Open your mind. LUT.
Lappeenranta University of Technology

LAPPEENRANTA UNIVERSITY OF TECHNOLOGY

Faculty of Technology

Department of Mathematics and Physics

Bankruptcy prediction using fuzzy methods

The topic of this Master's thesis was approved by the departmental council of the Department of Mathematics and Physics on February 7, 2012.

Examiners of this thesis were Ph.D Pasi Luukka and Ph.D Jouni Sampo.
The thesis was supervised by Ph.D Pasi Luukka.

Lappeenranta, March 22, 2012.

Onesfole Kurama
Technologiapuistonkatu A 12
53850 Lappeenranta
Mobile:+385465652944
onesfole@yahoo.com

ABSTRACT

Lappeenranta University of Technology

Faculty of Technology

Technomathematics

Onesfole Kurama

Bankruptcy prediction using fuzzy methods

Master Thesis

2012

73 pages, 18 figures, 18 tables, 1 appendix.

Supervisor: Ph.D Pasi Luukka.

Key words: Fuzzy Robust Principal Component Analysis(FRPCA), Similarity classifier, Fuzzy K-Nearest Neighbor classifier(FKNN), Bankruptcy prediction.

In this thesis, a classification problem in predicting credit worthiness of a customer is tackled. This is done by proposing a reliable classification procedure on a given data set. The aim of this thesis is to design a model that gives the best classification accuracy to effectively predict bankruptcy. FRPCA techniques proposed by Yang and Wang have been preferred since they are tolerant to certain type of noise in the data. These include FRPCA1, FRPCA2 and FRPCA3 from which the best method is chosen. Two different approaches are used at the classification stage: Similarity classifier and FKNN classifier. Algorithms are tested with Australian credit card screening data set. Results obtained indicate a mean classification accuracy of 83.22% using FRPCA1 with similarity classifier. The FKNN approach yields a mean classification accuracy of 85.93% when used with FRPCA2, making it a better method for the suitable choices of the number of nearest neighbors and fuzziness parameters. Details on the calibration of the fuzziness parameter and other parameters associated with the similarity classifier are discussed.

ACKNOWLEDGEMENTS

I wish to extend my sincere thanks and appreciation to the Department of Mathematics and Physics at Lappeenranta University of Technology, for the financial assistance accorded to me during my Masters degree programme. My genuine thanks should also reach Ph.D John Mango(Dean College of Natural Sciences) at Makerere University-Uganda, and Prof. Matti Heilio for enabling me to access CIMO scholarship which financed my studies for six months.

Great respect goes to my Supervisor Ph.D Pasi Luukka for all his tireless efforts, devotion, guidance and availing me with all the necessary materials. He is saluted also for all the course units that he taught me before we could produce this piece of work. I wish to thank Ph.D Jouni Sampo for sparing his material time to examine this thesis. In this regard, I would like to thank all the staff members at Lappeenranta University of Technology who taught me a variety of course units.

Special appreciation goes to my dear wife Orishaba Immaculate for her patience for all this time and tacking care of our daughter, Ronia Karungi. My Mother, brothers, sisters and a good number of friends have been very instrumental for my cause.
God bless you all.

March 22, 2012

Onesfole Kurama.

Contents

List of Figures

List of Tables

1. Introduction

The discovery of a fuzzy domain in mathematics by Prof. L. A. Zadeh(1965) [34] provided a fertile ground for very useful researches in a number of fields. It is now obvious that we cannot ignore our perceptions and limit ourselves to the traditional "Truth" or "False" to make viable decisions. A number of practical situations require more conclusions other than only true or false, for example in many games there is a need to have a "Win", "Loss" or "Draw". In daily life it is important to reward positive attempts towards given tasks and such rewards may take intermediate values other than the full values or nothing at all. Fuzzy mathematics provides the best way of assigning values according to performance on given tasks.

Applications of fuzzy methods to other fields have greatly taken shape since its discovery up to date with vast exciting results. Important areas using fuzzy mathematics include: industrial control, expert systems, decision analysis, software engineering, operation research, economics, medicine and many others. [15] In this thesis we have dealt with a financial application.

The central problem of this thesis is to predict bankruptcy for an applicant who needs a credit card. This is done by taking a thorough study of the underlying data obtained by credit card screening panels. It should be noted that with the world facing financial problems now and again, it is important for financial institutions to put up reliable measures to deal with financial matters.

Basically, it is a challenging task to carry out financial studies manually. It is therefore very helpful to design an automatic system or algorithm that can do such studies with ease, given the vast computer technology. It is clearly observed that data collected in the real field situation is always coupled with uncertainties, most of these uncertainties

or noise are a consequence of impreciseness and vagueness or sometimes they are caused by variability and ignorance. [2]

Australian credit card screening data set have been used in this thesis, it is obtainable for study purposes from Machine Learning Repository by Quinlan. [10] To describe this data briefly, it has 14 features or attributes and a class attribute, some of these attributes are numerical but others are categorical. The total number of instances is 690, which can be classified as positive or negative. However, 37 of these instances have some missing data but there are techniques to solve this problem. [10]

It remains a task to find out which dimension is suitable for proper studies with this data, we find this by applying dimension reduction techniques. [19] One of the popular techniques to reduce the dimension is the use of principal component analysis(PCA) as described by Jolliffe(1986). [11] In this method, new data is found by considering mutually orthogonal linear combinations of the original data. However, in our approach, fuzzy robust principal component analysis techniques are employed. These techniques were partially used by Luukka (2009) [19], for classification with the similarity classifier. We shall also use the fuzzy K-Nearest Neighbor [7] to further study our classifications.

The task of bankruptcy prediction has been a major concern for years not only to individuals but also to financial firms. A variety of approaches have been developed by several schoolers beginning with Bayesian techniques of minimizing the probability of error in estimation. [9] Kumar *et al* (2007) [16], described a comprehensive review of what has been done in this field since 1968-2005 with a variety of methods that have been employed to deal with the problem of bankruptcy. Basically the methods mentioned include: statistical techniques, neural networks, case-based reasoning, decision trees, operational research, evolutionary approaches, rough set-based techniques, support vector machine and others .

Atiya(2001) [1], used neural networks to solve the problem of bankruptcy, many other papers have been published using this approach. [26], [5] His method employed some techniques described in the paper by Zhang *et al*(1999) [35] where the neural network method is compared with the method of logistic regression. Apart from Bayesian approach and logistic regression which are old traditional ways of predicting errors in data analysis, the use of neural networks provided a new atmosphere in dealing with financial problems. [1]

Support vector machine(SVMs) approach have also been employed in bankruptcy prediction. Support vector machine with optimal choice of kernel function parameters approach was applied by Min & Lee(2005) [22] to a bankruptcy prediction problem and a new model improving the prediction accuracy was provided. In this direction, many other papers have been published with newer techniques involving artificial intelligence methods and better analysis approaches in the steps involved in analysis. [6], [22]

This thesis concentrates on creating linear combinations from the given data to form a new set of data and then applying fuzzy techniques in classification. The study is not interested in preliminary studies on the data, like histograms but is based on advanced studies involving feature extraction. Generally, it should be understood that for data analysis to be meaningful, there should be enough data to work with, otherwise it becomes trivial.

This thesis is organized as follows: In chapter two we present prerequisite theoretical concepts required in other chapters. Chapter three gives the approach used in this research, where we describe the main algorithms employed and the key theory. In chapter four, we present results obtained by using the described algorithms in the previous chapter. Chapter five discusses the effect of using different classifiers and finally the last chapter gives our general conclusions and future work.

2. Theoretical background

2.1 Preliminary concepts

The following notions and definitions should be at the finger tips of the reader before proceeding with the rest of the work in this thesis. Most of these shall be mentioned over and over without loss of generality.

2.1.1 Fuzzy set theory

The definition of a fuzzy set is basically an extension of the crisp case definition, that makes use of a characteristic function. [15] In the crisp case, a set A can be represented by a characteristic function m_A, which specifies which elements are in A and those that are not. [24]

$$m_A(x) = \begin{cases} 1, & if \quad x \in A \\ 0, & if \quad x \notin A \end{cases} \tag{1}$$

The fuzzy set case assigns values in the interval $[0, 1]$ to elements depending on their degree of membership to the concerned fuzzy set. [2] The membership degree is defined by using a suitable function.

Consider a fuzzy set A in Figure (1) below. Element a is fully inside A and its membership degree should clearly be 1. Element b is partially in A and it should have a membership degree less than 1. Element c is in the neighborhood of A, we notice that since A still has an influence in its neighborhood, element c should also have a non-zero membership degree. In principle, an element takes on zero if we have no information about it in rela-

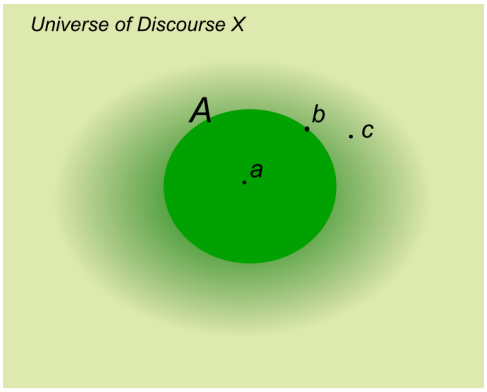


Figure 1: A fuzzy set A defined on the universe of discourse X.

tion to the concerned set.

The description above leads us to an acceptable definition of a fuzzy set as introduced by L. A. Zadeh. [34] Let X be a non-empty set. A fuzzy set A in X is characterized by a membership function

$$\mu_A : X \to [0, 1] \tag{2}$$

where $\mu_A(x)$ is interpreted as the degree of membership of element x in a fuzzy set A for each $x \in X$. However, we can always use the notation below when referring to the membership function without loss of generality. [2]

$$A : X \to [0, 1]$$

Clearly it is easier to write $A(x)$ than writing $\mu_A(x)$ all the time. There is a general representation of discrete fuzzy sets as far as the membership functions are concerned, introduced by Prof. L. A. Zadeh [34], which gives a unique conception. The membership degree is put as the numerator and the corresponding number as a denominator. Let X be a finite set with n elements, $X = \{x_1, x_2, x_3, ..., x_n\}$ and A be a fuzzy set defined on X, the membership function of the fuzzy set is written in discrete case as:

$$\mu_A(x) = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \mu_A(x_3)/x_3 + ... + \mu_A(x_n)/x_n \tag{3}$$

or simply

$$A(x) = A(x_1)/x_1 + A(x_2)/x_2 + A(x_3)/x_3 + ... + A(x_n)/x_n$$

The positive sign in equation (3) suggests the union as defined in set theory and the equation holds for the discrete case only. [15] For the continuous case, where X is an interval of real numbers, the membership function is got from:

$$\mu_A(x) = \int_X \mu_A(x)/x \tag{4}$$

In this case, the integral sign does not have its usual meaning, it only indicates that the pairs x and $\mu_A(x)$ in the interval X correctively form set A.

Fuzzy sets consist of elements and membership function values for their elements. In continuous case where elements belong to real intervals, there are several forms that a fuzzy set can take on, but the most common ones in practice are: the triangular shaped (Λ-shaped), trapezoidal (Π-fuzzy set) and the S- shaped fuzzy sets with membership functions as defined below:

Triangular shaped fuzzy set: For a continuous fuzzy set defined on areal interval $[\alpha, \gamma]$ as its universe of discourse, we have:

$$\Lambda(x : \alpha, \beta, \gamma) = \begin{cases} 0, & if \quad x \leq \alpha \\ \frac{x-\alpha}{\beta-\alpha}, & if \quad \alpha < x \leq \beta \\ \frac{\gamma-x}{\gamma-\beta}, & if \quad \alpha < x \leq \beta \\ 0, & if \quad x \geq \gamma \end{cases} \tag{5}$$

Trapezoidal fuzzy set: For a continuous fuzzy set defined on areal interval $[\alpha, \delta]$ as its universe of discourse, we have:

$$\Pi(x:\alpha,\beta,\gamma,\delta)=\begin{cases}0, & if \;\; x<\alpha\\ \frac{x-\alpha}{\beta-\alpha}, & if \;\; \alpha\leq x<\beta\\ 1, & if \;\; \beta\leq x<\gamma\\ \frac{\gamma-x}{\delta-\gamma}, & if \;\; \gamma<x\leq\delta\\ 0, & if \;\; x>\delta\end{cases} \tag{6}$$

S-shaped fuzzy set: For a continuous fuzzy set defined on areal interval $[\alpha,\gamma]$ as its universe of discourse, we have:

$$S(x:\alpha,\beta,\gamma)=\begin{cases}0, & if \;\; x\leq\alpha\\ 2(\frac{x-\alpha}{\gamma-\alpha})^2, & if \;\; \alpha<x\leq\beta\\ 1-2(\frac{x-\gamma}{\gamma-\alpha})^2, & if \;\; \beta<x\leq\gamma\\ 1, & if \;\; x>\gamma\end{cases} \tag{7}$$

It can be clearly observed that the highest membership value of an element in a fuzzy set is always 1, however, some sets may have all the membership values of their elements less than 1. This case brings in more definitions and notations, we shall briefly mention some of them so as to have a clear background for other essential concepts.

The **height** of a fuzzy set, hgt(A) is defined as the largest value of the membership values, [15] that is;

$$hgt(A)=\sup_{x\in U}\mu_A(x) \tag{8}$$

where sup means the supremum. If the height of a fuzzy set is 1, then the set is said to be **normal**, Otherwise it is sub-normal. [15] Elements of a normal fuzzy set with membership values equal to 1 form the **core** of this set [15], that is, the core of a fuzzy

set is the crisp subset of U such that $\mu_A(x) = 1$,

$$core(A) = \{x \in U : \mu_A(x) = 1\} \tag{9}$$

Elements of a fuzzy set whose membership values are non-zero, form the **support** of the set [15], that is, the support of a fuzzy set, supp(A) is defined as,

$$supp(A) = \{x \in U : \mu_A(x) > 0\} \tag{10}$$

Figure (2) below describes the concepts defined above.

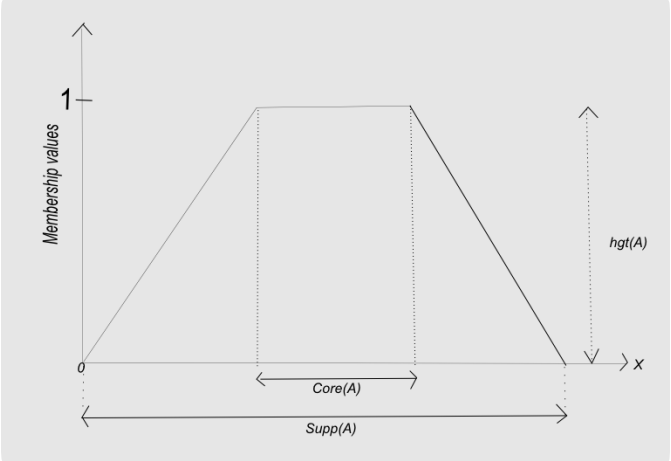


Figure 2: Core, Support and height of a fuzzy set A.

A more close term related to notations given in the definitions above is the concept of an $\alpha - cut$. [15] The $\alpha - cut$ of a fuzzy set A contains elements with membership values that have a value that is at least α, that is,

$$A_\alpha = \{x \in U : \mu_A(x) \geq \alpha\},\ \alpha \in [0, 1) \tag{11}$$

The concept of α-cuts is highly applicable in most operations with fuzzy sets and fuzzy numbers. They also provide a powerful tool to handle multiplication and division of fuzzy numbers which we encounter in daily applications.

A **convex** fuzzy set can also be defined using the general idea of convex sets, for any two elements x and y of a set [2], that is, for any $x, y \in X$ and $\lambda \in [0, 1]$ we have,

$$\mu_A(\lambda(x) + (1 - \lambda)y) \geq min\{\mu_A(x), \mu_A(y)\} \tag{12}$$

This formula implies that the fuzzy set represented by Figure (2) above is convex, where as the fuzzy set represented by the function in Figure (3) below is not convex.

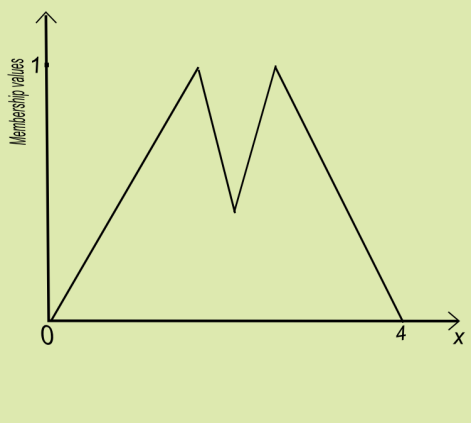


Figure 3: A fuzzy set that is not convex.

Next we define important operations on fuzzy sets that are also defined for crisp sets. In general, all operations on crisp sets are generalized to fuzzy set theory, these include complement, intersection, union and many others. We shall briefly define the standard intersection, union and complement because they are used in other sections of this thesis. However, it should be understood that they are not the only operations available for the fuzzy set theory. [15]

(i) Complement of a fuzzy set

In classical(crisp) set theory, the complement of a set is given by elements of the universal set that do not belong to the given set. In fuzzy set, the complement tells us how much do elements "not belong" to the given fuzzy set. For example, a set of "Rich

people" is a fuzzy set and its complement is the set of "Not rich people". The set of not rich people should tell us how much these people are not rich. [23]

In calculations we use the given membership values, say for a fuzzy set A, we get its complement A' from

$$\mu_{A'}(x) = 1 - \mu_A(x) \tag{13}$$

for all x in the Universe of discourse X. [23] Suppose a fuzzy set "Rich people" is given by,

$$Rich\ people = \{0.1/20 + 0.25/40 + 0.3/60 + 0.45/80 + 0.6/100 + 0.75/120\}$$

the complement of this set is got by employing the formula above, hence we get the fuzzy set of not rich people as,

$$Not\ rich\ people = \{0.9/20 + 0.75/40 + 0.7/60 + 0.55/80 + 0.4/100 + 0.25/120\}$$

Practical problems require a more clear definition of the complement as a function which assigns a value to each membership grade in a given fuzzy set. [15] This function should satisfy particular axioms like monotonicity and specified boundary conditions, but in other cases we may also need conditions like involution and continuity to hold.

(ii) Intersection of fuzzy sets

In fuzzy sets, the standard intersection or t-norm is got by considering the membership values that elements have in both sets. Suppose we have two fuzzy sets as: a set of "rich people, A" and "tall people, B", the intersection in the usual sense should be a set of people who are both rich and tall. However, for the fuzzy concern, the memberships of an element in these fuzzy sets may not necessarily be the same. Professor L. A. Zadeh [34] introduced a method that concentrates on the membership values and tells us how much of the element is in both sets. The standard intersection corresponds to the minimum of these membership values, but there are other t-norms that are frequently used

like the Algebraic product and the bounded difference. [23], [15] However, for calculation purposes, we shall use the formula,

$$\mu_{A\cap B}(x) = min[\mu_A(x), \mu_B(x)] = \mu_A(x) \cap \mu_B(x), \quad x \in U \tag{14}$$

Suppose we have two fuzzy sets defined by their membership values:

$$A = \{0.2/1 + 0.7/2 + 1/3 + 0.3/4 + 0.1/5\}$$

and

$$B = \{0.4/1 + 0.6/2 + 0.9/3 + 1/4 + 0.7/5\}$$

Intersection of these two sets would now be,

$$\mu_{A\cap B}(x) = \{0.2/1 + 0.6/2 + 0.9/3 + 0.3/4 + 0.1/5\}$$

The intersection, $i[\mu_A(x), \mu_B(x)]$, where i denotes the function for the intersection ($i(a, b) = min(a, b)$) is actually a binary operation on two unit intervals making sets A and B which produces another set also in the unit interval [15], thus,

$$i : [0, 1] \times [0, 1] \to [0, 1]$$

For the axioms of t-norm, refer to Klir and Yuan (1995). [15]
Solutions obtained in calculations depend on the choice of the t-norm used, it is therefore essential to specify which type of t-norm employed.

(iii) Union of fuzzy sets

The union of fuzzy sets defines how much of the element is in either set. [23] This is the dual of intersection as can be seen from the previous example: if one is rich then he is in the union set irrespective of whether he is tall or not. It basically means that the union is the largest set, thus for calculation purposes, the maximum operator is used. [15] Functions which behave like this are called t-conorms but to qualify to be t-conorms,

they must satisfy axioms like; boundary conditions, monotonicity, commutativity and associativity. However, if we meet any situation that requires the union, then the maximum operator shall be used. Given two fuzzy sets A and B, we define the union as:

$$\mu_{A\cup B}(x) = max[\mu_A(x), \mu_B(x)] = \mu_A(x) \cup \mu_B(x) \tag{15}$$

where $x \in U$, the universe of discourse. An example to demonstrate this idea is given next.

Example (2.1) Consider two fuzzy sets, "Rich people, A" and "tall people, B" defined by the membership values $A = \{0.4/1 + 0.2/2 + 1/3 + 0.7/4 + 0.2/5\}$ and $B = \{0.3/1 + 0.5/2 + 0.8/3 + 1/4 + 0.1/5\}$, find the union of the sets.
Solution: Applying the formula given above involving the use of maximum operator, the Union is got as:

$$\mu_{A\cup B}(x) = \{0.4/1 + 0.5/2 + 1/3 + 1/4 + 0.2/5\}$$

2.1.2 Fuzzy numbers

The innermost intuitive concept in fuzzy sets is probably that of fuzzy numbers because it provides a large platform to work with in this field. Fuzzy numbers are understood to be fuzzy subsets of the set of real numbers $\mathbb{R}$. [15] Fuzzy numbers are normally represented in a linguistic form, for example "approximately 4", "about 10" and many other words that lack exactness. Evidently, this kind of representation is used on a daily basis in almost all fields. The definition of a fuzzy number captures other properties and concepts defined for fuzzy sets like normality and convexity. [5]

Definition (2.1) A fuzzy number A is a normal, convex subset of a real line $\mathbb{R}$, whose membership function is a piecewise continuous function μ_A such that if $a \leq b \leq c \leq d$ are real numbers, the conditions below hold:
(i) $\mu_A(x) = 0$ for any $x \in (-\infty, a] \cup [d, \infty)$
(ii) $\mu_A(x)$ is increasing on the interval $[a, b]$ and decreasing on the interval $[c, d]$

(iii) $\mu_A(a) = 0,\ \mu_A(d) = 0$ and $\mu_A(x) = 1$ for all $x \in [b, c]$

A is a fuzzy number if $b = c$ and a fuzzy interval if $b < c$.

The geometry of a fuzzy interval as defined above is shown in Figure (4).

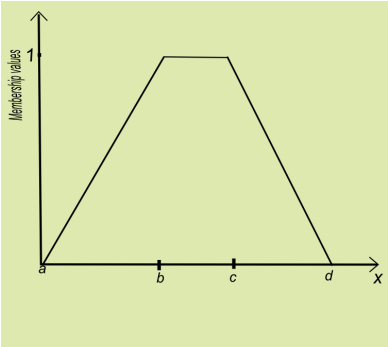


Figure 4: Description of a fuzzy interval.

Clearly, the membership function of a fuzzy number or fuzzy interval should preserve the mapping from $\mathbb{R}$ to the unit interval, that is,

$$\mu_A : \mathbb{R} \to [0, 1] \tag{16}$$

It is important to know how to find membership functions of fuzzy numbers since they are heavily used in practical problems involving computations. Suppose we have a fuzzy number represented by Figure (5). The membership function is got using equation (5) as $\mu_A(x)$, where:

$$\mu_A(x) = \begin{cases} 0, & if \quad x \leq 0 \\ \frac{x}{4}, & if \quad 0 < x \leq 4 \\ \frac{8-x}{4}, & if \quad 4 < x < 8 \\ 0, & if \quad x \geq 8 \end{cases}$$

The membership value of any element in the interval given above is obtained by substi-

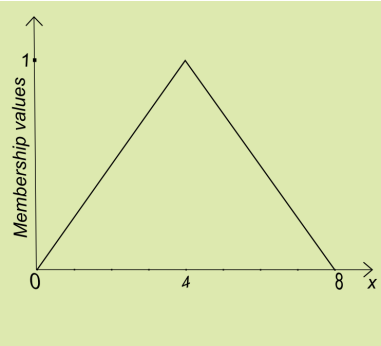


Figure 5: An example of a fuzzy number.

tuting in the function along side the interval, for example using the above example,

$$\mu_A(5.5) = (8 - 5.5)/4 = 0.625$$

Consider the fuzzy interval in Figure (6).

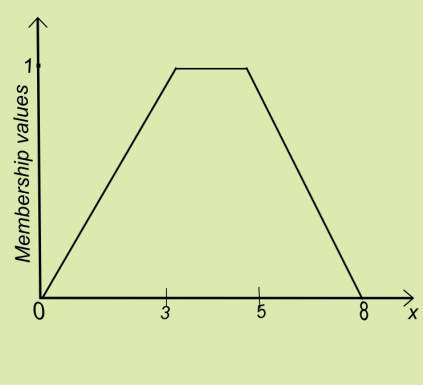


Figure 6: An example of a fuzzy interval.

The membership function is obtained by using equation (6), thus $\mu_A(x)$ is given by:

$$\mu_A(x) = \begin{cases} 0, & if \quad x < 0 \\ \frac{x}{3}, & if \quad 0 \leq x < 3 \\ 1, & if \quad 3 \leq x < 5 \\ \frac{5-x}{3}, & if \quad 5 \leq x \leq 8 \\ 0, & if \quad x > 8 \end{cases}$$

It is easy to calculate α-cuts when one knows the membership function of a fuzzy number. The core and support of a fuzzy set can also be determined, from the above example, $core(A) = [3,\ 5]$ and $supp(A) = (0,\ 8)$

Operations of addition, subtraction, multiplication and division are applied in fuzzy numbers. One way of doing this is by calculating their α-cuts and applying interval arithmetics. For this topic we refer to Klir and Yuan. [15]

2.2 Fuzziness measures

The concept of measure in data analysis is vital because it provides one with a way to categorize which sets are sharper than others. [24] This gives an insight on how to relate and determine by calculation these values. At some point we need to see the effect of changing the fuzziness value and calibrate which fuzziness value is best for the given data. Notice that the concept of cardinality for fuzzy sets is introduced here.

Consider a discrete fuzzy set A with a finite support, its cardinality (cardA) is obtained by summing the membership grades. [2] Thus, we have:

$$cardA = \sum_{x \in U} \mu_A(x) \tag{17}$$

where U is the Universe of discourse on which A is defined.

The relative cardinality($card_{rel}A$) is found for a finite universal set U as:

$$card_{rel}A = \frac{cardA}{cardU} \tag{18}$$

The next example shows how to compute cardinality of discrete fuzzy sets.

Example (2.2)

Define U as the set $U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ and A as a fuzzy set defined by approximating its membership functions by $\mu_A(5) = 0.2$, $\mu_A(1) = 0.1$, $\mu_A(2) = 0.6$, $\mu_A(3) = 1$, $\mu_A(4) = 0.5$. Calculate the cardinality of set A in the given universal set U and the relative cardinality.

Solution:

Using the formulas given above, we need to sum the membership values to get the cardA.

$$cardA = \sum_{x \in U} \mu_A(x)$$

$$= 0.1 + 0.6 + 1 + 0.5 + 0.2 = 2.4$$

The cardinality of U is given by,

$$cardU = 1 + 1 + 1 + 1, ..., +1 = 10$$

Hence the relative cardinality is given by,

$$card_{rel}A = \frac{cardA}{cardU} = \frac{2.4}{10} = 0.24$$

In real situations, there are many cases where the the universal set is infinite, as well as continuous fuzzy sets, for such cases we cannot employ the formulas given above. If the fuzzy set A is contained in some compact set inside U, the integration technique is used to find the cardinality as, [2]

$$cardA = \int_U \mu_A(x)dx \tag{19}$$

The relative cardinality for this case relies on defining a weighting function which should scale the membership functions in the given area. [2] It is required that the weighting function $p(x)$ sum to 1, thus it can be defined such that,

$$\int_U p(x)dx = 1$$

When this condition is met, then the cardinality with respect to p is found from the formula [24],

$$card_{rel(p)}A = \int_U \mu_A(x)p(x)dx \tag{20}$$

Next we introduce the fuzziness measure. Precisely speaking, a fuzziness measure on a set U is a mapping $p : F(U) \to [0, \infty)$ so that we define an ordering $\prec_p$ on $F(U)$ by taking note of the following: [2] There are sharpest sets made of minimum elements say A_0 such that $p(A_0) = 0$ and for any two fuzzy sets A and B, we have

$$A \prec_p B :\Leftrightarrow p(A) \leq p(B) \tag{21}$$

where $A \prec_p B$ is interpreted as "A is sharper than B with respect to p" [2].

The most common fuzziness measure that we encounter in computations is the entropy measures, this is also called the narrow-sense fuzziness measure h. [24] This is evaluated such that any crisp set A_0 leads to $h(A_0) = 0$.

Definition (2.2)

A non-negative function h on $F(U)$ is called a narrow sense fuzziness measure(entropy measure) if and only if the conditions below hold:

(i) $h(A) = 0$ if A is an ordinary set.

(ii) $h(A)$ is maximum if $\mu_A(x) = 0.5$

(iii) if for any $x \in U : [\mu_B(x) < 0.5 \wedge \mu_A(x) \leq \mu_B(x)] \vee [\mu_B(x) > 0.5 \wedge \mu_A(x) \geq \mu_B(x)]$,then $h(A) \leq h(B)$. [2]

De Luca(1972) [8], proposed the simplest method of computing the entropy which makes use of the expression below:

$$h(A) = -\sum_{x \in U} \mu_A(x) ln(\mu_A(x)) + (1 - \mu_A(x)) ln(1 - \mu_A(x)) \tag{22}$$

The next example uses De Luca's method.

Example (2.3)

Define a fuzzy set A which is approximated by the membership values given as $\mu_A(6) = 0.99$, $\mu_A(7) = 0.9$, $\mu_A(8) = 0.7$, $\mu_A(9) = 0.5$, $\mu_A(10) = 0.1$ Calculate the entropy measure (the narrow-sense fuzziness measure) using De Luca's method.

Solution:

Notice that $x = \{6, 7, 8, 9, 10\}$ and $\{0.99, 0.9, 0.7, 0.5, 0.1\}$. From De Luca's expression, we only need to calculate the natural logarithms and sum over all the values. That is,

$$h(A) = -\sum_{x \in U} \mu_A(x) ln(\mu_A(x)) + (1 - \mu_A(x)) ln(1 - \mu_A(x))$$

$h(A) = -0.99 ln(0.99) - (1 - 0.99) ln(1 - 0.99)$
$- 0.9 ln(0.9) - (1 - 0.9) ln(1 - 0.9)$
$- 0.7 ln(0.7) - (1 - 0.7) ln(1 - 0.7)$
$- 0.5 ln(0.5) - (1 - 0.5) ln(1 - 0.5)$
$- 0.1 ln(0.1) - (1 - 0.1) ln(1 - 0.1)$
≈ 2.0102

Another approach to calculate fuzziness was proposed by Yager(1979) [19]. It makes use of the maximum element of the ordering defined by h, this is actually an $\alpha - cut$, for $\alpha = 0.5$. The performance of the expression depends on the fuzziness variable n, thus Yager's equations is given as:

$$h(A) = [\sum_{x \in U} |\mu_A(x) - \mu_{A_{0.5}}(x)|^n]^{\frac{1}{n}}, \quad n \in [1, \infty) \tag{23}$$

According to Yager(1979) [19], for a fuzzy set A, the size of $A \cap A'$ can perfectly be used to calculate the deviation of fuzzy set A from its crisp case. The given model simplifies when $n = 1$, the result of this is got by computing the height or cardinality. That is,

$$h(A) = hgt(A \cap A') \tag{24}$$

which is estimated to be the same as:

$$h(A) = card(A \cap A'). \tag{25}$$

The next example demonstrates the use of Yager's method to calculate fuzziness for simple cases where $n = 1$, we end up with the index of fuzziness. [2]

Example (2.4)

Let A be a fuzzy set "approximately 8" whose elements have the following membership values, $\mu_A(6) = 0.4,\ \mu_A(7) = 0.6,\ \mu_A(8) = 1.0,\ \mu_A(9) = 0.5,\ \mu_A(10) = 0.1$. Use Yager's method to calculate the fuzziness measure for this ordinary set A.

Solution:

From above we have,

$$A = 0.4/6 + 0.6/7 + 1.0/8 + 0.5/9 + 0.1/10$$

The complement is given by:

$$A' = 0.6/6 + 0.4/7 + 0.0/8 + 0.5/9 + 0.9/10$$

Next we calculate $A \cap A'$ using the minimum operator

$$\mu_{A \cap A'} = min\{\mu_A(x), \mu_{A'}(x)\}$$

we get,

$$A \cap A' = 0.4/6 + 0.4/7 + 0.0/8 + 0.5/9 + 0.1/10 \approx 1.40$$

Notice that the solution obtained in example (2.4) is influenced by the kind of the fuzzy complement used.

Another important type of fuzziness measure for us to mention in this study is the energy measure, e. [2] This type of measure is important because it evaluates deviations of the fuzzy set from the empty set, that is, we consider an initial set as $A_0 = \emptyset$. The corresponding energy measure is then given by $e(\emptyset) = 0$. [24] The fact that the energy measure is got by deviations from an empty set makes all non-empty sets to have positive measures of this kind. Hence we shall always expect that for any set A such that $A \neq \emptyset$ we have $e(A) > 0$. [2]

Specifically speaking, the height of a fuzzy set A, $hgt(A)$ is a good example of the energy measure. There are some cardinalities and relative cardinalities that are examples of the energy measure due to its principle of calculation from the empty set. Notice here that if h is a narrow-sense fuzziness measure or entropy measure of a fuzzy set A, then $e(A) = h(A \cap A')$ is also an energy measure. At this point we are now comfortable to define an energy measure.

Definition (2.3)

A non-negative function e on $F(U)$ is called an energy measure if and only if,

(i) $e(\emptyset) = 0$

(ii) If for any $x \in U$, it holds that $\mu_A(x) \leq \mu_B(x)$ this is the same as $e(A) \leq e(B)$. [24]

It should be noted that we can find both the entropy measure and energy measure for any fuzzy set even if it is not normal. This is opposed to the measure of fuzziness called measure of non-specifity which works only when the concerned fuzzy set A is normal, $(hgt(A) = 1)$. [2] We shall not discuss measure of non-specifity in details here but its an important measure when specifying impreciseness or vagueness where deviations are measured from one-point sets.

2.3 Fuzzy relations

In this section, general ideas on classical relations are extended to fuzzy relations. The concept of relations is meaningful, if it is able to explain the presence or absence of interaction between elements of two or more sets. [15] A classical relation R is considered as a subset, containing ordered pairs from interacting crisp sets $A, B, ..., Z$. [2] A fuzzy relation, is thus defined on the cartesian product of crisp sets but with elements having membership degrees in $[0, 1]$. In analysis of practical problems, relations or associations among objects are very crucial and unavoidable. Essentially, there are several types of relations that have been extensively studied with a variety of important properties for example order relations and equivalence relations. [24]

2.3.1 Binary fuzzy relations

Let $A_1, A_2, ..., A_n$ be crisp sets, the cartesian product $A_1 \times A_2$ forms a binary subset. In the same way, subsets formed by $A_1 \times A_2 \times ... \times A_n$ are n-ary. [5], [2] Any classical relation R is defined by a characteristic function which assigns a value 1 or 0 to elements or tuples depending on whether they belong to the relation or not respectively. Thus for a crisp relation, we have:

$$A_R(u, v) = \begin{cases} 1, & if \quad (u, v) \in R \\ 0, & if \quad (u, v) \notin R \end{cases} \tag{26}$$

For a binary fuzzy relation R on sets A_1 and A_2, elements of the tuple $(u, v) \in R$ have membership degrees in $[0, 1]$ showing the degree of relationship between u and v for $u \in A_1, v \in A_2$. [2] In this way, $A_R(u, v)$ or simply $R(u, v)$ is interpreted as the degree of membership of the ordered pair (u, v) in a binary relation R. [5]

Next we give an example of a binary fuzzy relation on a single set A to show how the

membership function of the fuzzy relation R should be written.

Example (2.5)

Define a binary fuzzy relation R as "approximately equal" on a set $A = \{1, 3, 5\}$ by $R(1,1) = R(3,3) = R(5,5) = 1$, $R(1,3) = R(3,1) = R(3,5) = R(5,3) = 0.6$, and $R(1,5) = R(5,1) = 0.4$.

The membership function of the fuzzy relation R can be written as:

$$R(x,y) = \begin{cases} 1 & if \quad |x-y| = 0 \\ 0.6 & if \quad |x-y| = 2 \\ 0.4 & if \quad |x-y| = 4 \end{cases}$$

In applications involving operations on fuzzy relations, membership grades can be written in matrix form for simplicity. Entries of the matrix are clearly composed of relationships between elements of given sets. From example (2.5) above, we could write $R(x,y)$ in matrix form as:

$$R(x,y) = \begin{bmatrix} 1 & 0.6 & 0.4 \\ 0.6 & 1 & 0.6 \\ 0.4 & 0.6 & 1 \end{bmatrix}$$

Given two sets A, B and a fuzzy relation $R(x,y)$, for all $x \in A$ and $y \in B$. The inverse of R is denoted by R^{-1}, where:

$$R^{-1}(y,x) = R(x,y) \tag{27}$$

Notice that for membership grades written in matrix form, the matrix representation of $R^{-1}(x,y)$ is the transpose of that one for $R(x,y)$ for the given fuzzy relation. [15]

Let $R(x, y)$ represent the extent to which element x is similar to y for all $(x, y) \in A \times A$. Clearly $R(x, y)$ takes on values within the unit interval [0, 1], thus we define some of the important properties for a fuzzy equivalence relation.

1. Reflexive if $R(x, x) = 1$
2. Symmetric if $R(x, y) = R(y, x)$
3. Transitive if $R(x, z) \geq R(x, y) \wedge R(y, z)$, where $\wedge$ stands for minimum. [24]

A fuzzy relation which is reflexive, symmetric and transitive is called a fuzzy equivalence relation or simply a similarity relation. [15] Great importance is attached to the equivalence class of a fuzzy relation due to its use especially in defining fuzzy partitions, for more on this topic see. [15] Other properties related to the three defined above are also extended to the fuzzy case without loss of generality. These include: irreflexive, anti-reflexive, asymmetric, antisymmetric and anti-transitive. For details on binary fuzzy relations, we refer to Klir and Yuan. [15]

2.3.2 Basic operations on fuzzy relations

A basic but one of the most important operations on fuzzy relations is that of standard composition. Let R and S be two binary fuzzy relations on sets given by the cartesian products $A_1 \times A_2$ and $A_2 \times A_3$ respectively. The standard composition on $R(A_1, A_2)$ and $S(A_2, A_3)$ is denoted by $R(A_1, A_2) \circ S(A_2, A_3)$. This gives a binary fuzzy relation $P(A_1, A_3)$ on the cartesian product $A_1 \times A_3$ defined by

$$P(x, z) = [R \circ S](x, z) = \max_{y \in A_2} min[R(x, y), S(y, z)] \tag{28}$$

for all $x \in A_1$ and all $z \in A_3$. This is normally called the max-min composition and is an essential tool in practical computations. [15] The next example shows how to compute the max-min composition.

Example (2.6) Consider two membership matrices of binary fuzzy relations defined

below, find the max-min composition.

$$\begin{bmatrix} 0.2 & 0.8 & 0.4 \\ 0.1 & 0.4 & 0 \\ 0.5 & 0.3 & 0.7 \end{bmatrix} \circ \begin{bmatrix} 0.5 & 1 & 0.4 \\ 0 & 0.7 & 0.6 \\ 0.9 & 0.1 & 0.2 \end{bmatrix}$$

Solution: Using equation (28),

$$r_{11} = \max[\min(.2,.5),\min(.8,0),\min(.4,.9)] = 0.4$$
$$r_{12} = \max[\min(.2,1),\min(.8,.7),\min(.4,.1)] = 0.7$$
$$\bullet$$
$$\bullet$$
$$\bullet$$
$$r_{33} = \max[\min(.5,.4),\min(.3,.6),\min(.7,.2)] = 0.4$$

The matrix below is obtained.

$$\begin{bmatrix} 0.4 & 0.7 & 0.6 \\ 0.1 & 0.4 & 0.4 \\ 0.7 & 0.5 & 0.4 \end{bmatrix}$$

The general operation of which the standard or max-min composition is a particular case is the sup-$i$composition, where i is a particular t-norm. [15] Replacing the minimum operator by i we define the sup-$i$composition by the equation

$$[R \circ S](x,z) = \sup_{y \in A_2} i[R(x,y), S(y,z)] \tag{29}$$

for all $x \in A_1$ and all $z \in A_3$, where the binary fuzzy relations are defined as earlier. There are many other operations on fuzzy relations like projection and cylindrical extensions, for all these we refer to Klir and Yuan. [15]

2.4 Similarity Measures

In this subsection, a background of what shall be discussed in similarity classifiers is put forward. The main concept behind similarity measures is fuzzy relations. The major challenge that is occasionally faced in such a study is the fact that not all data is simple to be compared. However, if the available data can be grouped, then general procedures can be utilized to obtain meaningful results.

Given two elements x and y, fuzzy similarity considers the degree of similarity that an element x of a fuzzy set, A has with element y. Hence, for each $x \in A$ we define the similarity class as a fuzzy set in which the membership grade of any particular element represents the similarity of that element to a selected element y. [2]

In practice, we deal with large data sets with many classes or sample points, for such cases we calculate the mean vectors for each class across all features. The required similarities are computed basing on these mean vectors in relation to the sample points in the data. [2] The general idea is that the similarity between an element and itself gives 1, thus the similarity between two different elements should be less than 1. One of the most important theory used in this thesis is based on Łukasiewicz similarity .[21]

Generalized Łukasiewicz similarity uses the similarity relation for two different elements x and y, written as:

$$x \Leftrightarrow y = \sqrt{(1 - |x^p - y^p|)^{\frac{1}{p}}}, \quad p \in [1, \infty) \tag{30}$$

This expression can be extended to cater for any required dimensions. Suppose we have an n-dimensional data with some particular vectors $x = \{x_1, x_2, ..., x_n\}$ and $y = \{y_1, y_2, ..., y_n\}$, then equation (30) extends to,

$$S(x, y) = \frac{1}{n} \sum_{i=1}^{n} (1 - |\mu_i(x)^p - \mu_i(y)^p|)^{\frac{1}{p}}, \quad p \in [1, \infty) \tag{31}$$

The next example shows the use of equation (31) for small values of p.

Example (2.7)

Let $x = [0.1, 0.9, 0.7, 0.3, 0.4]$ and $y = [0.25, 1, 0.5, 0.3, 0.75]$. Find the similarity $S(x, y)$ using similarity based on generalized Łukasiewicz similarity for the values of $p = [1, 2]$

Solution: For $p = 1$,

$$S(x,y) = \frac{1}{5}(1-|0.1-0.25|+1-|0.9-1|+1-|0.7-0.5|+1-|0.3-0.3|+1-|0.4-0.75|) = 0.8400$$

For $p = 2$,

$S(x,y) = \frac{1}{5}[((1-|0.1^2-0.25^2|)^{\frac{1}{2}}+(1-|0.9^2-1^2|)^{\frac{1}{2}}+(1-|0.7^2-0.5^2|)^{\frac{1}{2}}+(1-|0.3^2-0.3^2|)^{\frac{1}{2}}$
$+ (1 - |0.4^2 - 0.75^2|)^{\frac{1}{2}})] \quad = 0.9036$

3. Research approach

In this chapter, a brief description of the key directions taken in this thesis are discussed. Algorithms that have been used to obtain results are also presented for the reader to have a feeling of the theoretical source of our results. Much emphasis is put on fuzzy robust principal component analysis as well as fuzzy k-nearest neighbor approaches. We start with the general view of principal component analysis.

3.1 Principal component analysis

Data analysis becomes a challenge as the dimension increases, the increased number of measured features makes it hard to make proper studies and conclusions as well. However, it is again true that not all these measured features probably contribute largely to the results. It is therefore a genuine task to identify which features are so important to the study so that they can be dealt with closely and others possibly combined or removed. Principal component analysis(PCA) is one of the statistical techniques which can be used for dimension reduction. [11] It involves the formation of linear combinations from feature vectors which results in less dimension.

In this thesis we shall use fuzzy robust principal component analysis algorithms so it is important to briefly describe what PCA actually does. Different fields refer to PCA methods differently, in algebra it is related to singular value decomposition(SVD) and the procedures cut across even in other areas of research. [14] The statistical approach to this technique involves the use of statistical tools like mean and variance, then algebra is used in computing eigenvalues and eigenvectors to form the covariance matrix. We shall briefly describe the process below for possibly a large dimensional data set.

Suppose we have an r-dimensional vector of measured features, $x = (x_1, x_2, ..., x_r)^T$ and

we would like to reduce the dimension to k or $y = (y_1, y_2, ..., y_k)^T$ where $k < r$. It actually means that we have say an $n \times r$ order that we want to reduce to $n \times k$. The first key task is to center this data by use of the mean of the vectors down the column such that the mean of the adjusted data is zero [11] [14], that is,

$$\frac{1}{n}\sum_{i=1}^{n} x_i = 0 \tag{32}$$

Covariances for the r-dimensional data are calculated and written in a matrix form. It is evident that the order of this covariance matrix depends on the dimension of the data and for a large data, many covariances need to be calculated. Hence, the task of coming up with a covariance matrix should be accomplished via techniques like computation of projections.

Principal component analysis procedures aim at finding a subspace M of R^k, such that the orthogonal projections $P_M x_i$ of the r values on the selected subspace M have a maximal variance. [11] Due to the technical adjustment that is made using the mean, the vectors making the covariance matrix are unit vectors. Vectors with largest variance values form the principal components to be extracted from the data. Suppose v is a unit vector spanning the subspace M and v' its transpose, then we get the projection of $x \in R^k$ on M as,

$$P_M x = (v'x)v \tag{33}$$

The computed variance of the data in the direction of M is then given by the formula below: [2]

$$\frac{1}{n}\sum_{i=1}^{n}(v'x_i)^2 = \frac{1}{n}\sum_{i=1}^{n} v'x_i x_i' v = v'(\frac{1}{n}\sum_{i=1}^{n} x_i x_i')v = v'Sv \tag{34}$$

where S is the matrix of all the covariances computed from the data. The PCA algorithm will therefore extract the vector which will try to maximize $v'Sv$ over all the unit vectors

as constraints. [11] The largest eigenvalue say α_1 provides the corresponding required solution in the vector form say v_1 such that we have

$$v_1' S v_1 = \alpha_1 v_1' v_1 = \alpha_1 \tag{35}$$

This idea is used to extract all the required k-dimensional subspace M where the projections computed will have the largest variance. Principal axes are obtained as lines spanned by the computed eigenvectors, from which the required principal components forming the new data are got. [11] [17] We shall keep referring to this procedure as the traditional way of obtaining a new data, where the new data represents the old data without loss of generality.

Generally, principal component analysis techniques are relatively sensitive to outliers in the data. This makes the method rather incompetent when dealing with noisy data. However, PCA techniques have effectively been used in neural networks to compute weights of the so-called hidden nodes. [26] This tool is actually very important and for such application one may only need to find the eigenvector associated to the most significant eigenvalue but not to compute all eigenvectors as it is done in the PCA procedure presented above.

3.2 Fuzzy robust principal component analysis

In real data, outliers normally exist due to several reasons, it could be a result of machine errors or even human errors done in recording. It is a great deal if a method to eliminate these outliers is discovered. Fuzzy robust principal component analysis algorithms are designed to detect outlier so that they are removed before weight computations are done. This makes FRPCA techniques essential and better accuracies are yielded when these algorithms are used. In this thesis we have used FRPCA algorithms proposed by Yang and Wang. [32]

The algorithms proposed by Yang and Wang in their paper are based on the classical(crisp) case given by Xu and Yuille. [30] The ability of these algorithms lay in the way they deal with outliers removal in a given data. The same algorithms have been used by Luukka [17] in classification involving Breast cancer data set and similarity classifier. Yang and Wang [32] proposed that the function to be minimized can be extended to be fuzzy from the usual crisp objective function given by Xu and Yuille. [30] We present a summary description of the theory by Xu and Yuille [30] and the modification proposed by Yang and Wang. [32]

Xu and Yuille [30] proposed an optimization function with an energy measure $\theta(x_i)$, subject to the membership set $v_i \in \{0, 1\}$ given as:

$$H(V, \phi) = \sum_{i=1}^{k} v_i \theta(x_i) + \lambda \sum_{i=1}^{k} (1 - v_i) \tag{36}$$

where $V = \{v_i : i = 1, 2, 3, ..., n\}$ is the membership set, $X = \{x_1, x_2, x_3, ..., x_n\}$ is the data set that one is using and λ is the threshold. The optimization function (36) should be minimized with respect to v_i and ϕ but the major challenge is that v_i is discrete while ϕ is a continuous variable. This poses a technical challenge to optimization procedures as a result of different domains. To overcome this, Xu and Yuille [30] transformed the minimization problem into a maximization problem of the Gibbs distribution with the use of a partition function. The new problem thus looks like below:

$$S(V, \phi) = \frac{exp(-\xi H(V, \phi))}{R} \tag{37}$$

where R is the partition function that should scale the maximization problem such that:

$$\sum_{V} \int_{\phi} S(V, \phi) = 1. \tag{38}$$

Special treatment of the energy measure was also done by Xu and Yuille. [30] Notice that there are several kinds of measures but for this problem, they took the measure $\theta(x_i)$ to

be any of the functions $\theta_1(x_i)$ or $\theta_2(x_i)$ given below:

$$\theta_1(x_i) = \|x_i - \phi^T x_i \phi\|^2 \tag{39}$$

$$\theta_2(x_i) = \|x_i\|^2 - \frac{\|\phi^T x_i\|^2}{\|\phi\|^2} = x_i^T x_i - \frac{\phi^T x_i x_i^T \phi}{\phi^T \phi} \tag{40}$$

The minimization should be done over all values and the best rules for minimizing the functions $H_1 = \sum_{i=1}^{k} \theta_1(x_i)$ and $H_2 = \sum_{i=1}^{k} \theta_2(x_i)$ were given as:

$$\phi^{new} = \phi^{old} + \alpha_t[y(x_i - v) + (y - u)x_i] \tag{41}$$

$$\phi^{new} = \phi^{old} + \alpha_t(x_i y - \frac{\phi}{\phi^T \phi} y^2) \tag{42}$$

where $y = \phi^T x_i,\ v = y\phi,\ u = \phi^T v$ and parameter α_t is the learning rate. [30], [17]

A general modification was made on equation (36) by Yang and Wang [32] by altering the membership set v_i with a factor called the fuzziness variable denoted by n in the next equation below, thus the objective function was stated as:

$$H = \sum_{i=1}^{k} v_i^n \theta(x_i) + \lambda \sum_{i=1}^{k} (1 - v_i)^n \tag{43}$$

subject to $v_i \in [0, 1]$ and $n \in [0, \infty)$. In practical experiments it is important to tune n so that one gets the best value for the fuzziness variable which gives best classification values. The beauty of this result is that we now have v_i continuous as opposed to the earlier case where it was discrete, so the gradient descent method can be used. The derivatives of (43) with respect to both v_i and ϕ were found:

$$\frac{\delta H}{\delta v_i} = \frac{\delta}{\delta v_i}(\sum_{i=1}^{k} v_i^m \theta(x_i) + \lambda \sum_{i=1}^{k} (1 - v_i)^n) \tag{44}$$

Setting the derivative to zero gives the solution as:

$$v_i = \frac{1}{1 + (\theta(x_i)/\lambda)^{\frac{1}{(n-1)}}} \tag{45}$$

Using this result in the objective function and simplifying, we obtain:

$$H = \sum_{i=1}^{k} (\frac{1}{1 + (\theta(x_i)/\lambda)^{\frac{1}{(n-1)}}})^{n-1} \theta(x_i) \tag{46}$$

Doing the same procedure with respect to ϕ we obtain:

$$\frac{\delta H}{\delta \phi} = (\frac{1}{1 + (\theta(x_i)/\lambda)^{\frac{1}{(n-1)}}})^{n} (\frac{\delta \theta(x_i)}{\delta \phi}) \tag{47}$$

For easy analysis, of the expression above, let

$$\beta(x_i) = (\frac{1}{1 + (\theta(x_i)/\lambda)^{\frac{1}{(n-1)}}})^{n} \tag{48}$$

we end up with

$$\frac{\delta H}{\delta \phi} = \beta(x_i).(\frac{\delta \theta(x_i)}{\delta \phi})$$

Importance is attached to the fuzziness variable n but there is no general rule to tune this value, however, Yang and Wang [32] preferred $n = 2$. Looking at equation (45), if $n = 1$ the fuzzy membership reduces to the hard membership (see [30]) and can be calculated using the function:

$$v_i = \begin{cases} 1, & if \quad e(x_i) < \lambda \\ 0, & otherwise \end{cases} \tag{49}$$

where λ is the hard threshold. In this thesis we have done preprocessing of our data to have it in a more feasible way by using these FRPCA algorithms.

Steps followed in implementing FRPCA algorithms are listed next. Notice that there are three slightly different algorithms but we have applied all of them in obtaining our results as a way to see which one is best. Elaborate procedures can be found in. [32]

FRPCA1 algorithm:

Step 1 Initially set the iteration count to $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$, soft threshold λ to a small positive value and randomly initialize the weight ϕ.

Step 2 While t is less than T, perform the next steps 3 to 9.

Step 3 Compute $\alpha_t = \alpha_0(1 - t/T)$, set $i = 1$ and $\sigma = 0$.

Step 4 While i is less than n perform steps 5 to 8.

Step 5 Compute $y = \phi^T x_i,\ v = y\phi$ and $u = \phi^T v$.

Step 6 Update the weight with, $\phi^{new} = \phi^{old} + \alpha_T \beta(x_i)[y(x_i - v) + (y - u)x_i]$.

Step 7 Update the temporary count $\sigma = \sigma + \theta_1(x_i)$.

Step 8 Add 1 to i.

Step 9 Compute $\lambda = (\sigma/n)$ and add 1 to t.

FRPCA2 algorithm:

The procedures above are followed except step 6 and 7 where we change the way we update the weight and count.

Step 6 Update the weight with, $\phi^{new} = \phi^{old} + \alpha_T \beta(x_i)(x_i y - \frac{\phi}{\phi^T \phi} y^2)$.

Step 7 Update the temporary count $\sigma = \sigma + \theta_2(x_i)$.

FRPCA3 algorithm:

Follow the same procedure as FRPCA1 except step 6 and 7 .

Step 6 Update the weight with, $\phi^{new} = \phi^{old} + \alpha_T \beta(x_i)(x_i y - \phi y^2)$.

Step 7 Update the temporary count $\sigma = \sigma + \theta(x_i)$.

3.3 Pattern recognition

3.3.1 Introduction

Statistical problems have for a long time required special techniques when it comes to decision-making. A number of mathematical tools have been developed over years for this cause but pattern recognition is central to the whole idea. The general aim of pattern recognition is to carry out classification of not only numbers; but also objects which could be images or signal waveforms. [29] This makes pattern recognition a very broad chapter with applications in many other fields like industry and other aspects of technology like machine intelligence systems. [12]

Improved technology have greatly twisted this topic with ideas on how to make automatic machines and robots to do work in a cheaper way or even to perform in sections where human beings could face a danger of radiation exposures. [29] It is believed that decision-making abilities of a person are greatly related to recognizing patterns associated with what he is challenged to work with, for example learning a new language is based on the pattern of its alphabet or characters. In most cases pattern recognition is based on a very complex brain mechanism that we cannot manage to describe for now, due to this reason we shall only consider cases which do not pose such challenges.

In this thesis, we are interested in carrying out classification as accurate as possible using the known pattern recognition techniques. Two different classifiers have been used where their choice depended on the nature of the data set that we dealt with. Outliers in the data are removed by the robust nature of the FRPCA algorithms used and linear combinations formed which are used as new data.

3.3.2 Fuzzy clustering

Cluster analysis is a key aspect in pattern recognition that has been developed over years to study data structures. The general idea involved in cluster analysis is to locate the mean position or centers for each cluster that properly represents that particular cluster in the given data. When cluster centers are identified, it is possible to use them to allocate each object of the data to its respective cluster with greater accuracy. [15], [12], [29] It is required that each object be allocated to one and only one cluster but making such partitions is not a trivial task. The best idea that takes into account the fuzzy aspect of the data is to come up with a partition matrix that is comprised of membership values of each object under assignment. [2]

Fuzzy clustering in general involves replacing the usual classical(hard or crisp) partitioning with a somewhat weaker notion of a fuzzy partition. There are several ways of coming up with fuzzy partitions but the most used technique involves the use of center means and due to this, it is called the *fuzzy c-means clustering method*. [3], [15] The background of the fuzzy domain by L. A. Zadeh [34] acts as a basis for key suggestions by other scholars, for example a fuzzy clustering of a data set X into m clusters is characterized by m functions μ_i where,

$$\mu_i : X \to [0,1], \quad i = 1, 2, ..., m \tag{50}$$

such that we have,

$$\sum_{i=1}^{m} \mu_i(x_j) = 1, \quad j = 1, 2, ..., n \tag{51}$$

and

$$0 < \sum_{j=1}^{n} \mu_i(x_j) < n, \quad i = 1, 2, ..., m \tag{52}$$

The above fuzzy membership functions suggest a characterization that each vector x be-

longs to more than one cluster with some degree specified by the corresponding value of $\mu_i(x_j)$ which must sit in the unit interval $[0, 1]$. [29] In this context, values close to one show a high sense of belonging to that cluster and values close to zero show less chance of being in that cluster. [29]

The major challenge is how to create the partitions so that equations (50),(51) and (52) are satisfied. Bezdek [3] proposed a criterion that is helpful in finding cluster centers and then associate them with the way the criterion generally performs. Thus, he defines a quantity called performance index $J_m(p)$ which is given by:

$$J_m(p) = \sum_{j=1}^{n} \sum_{i=1}^{N} [\mu_i(x_j)]^m \|x_j - v_i\|^2 \tag{53}$$

where $\|x_j - v_i\|^2$ represents the distance between x_j and v_i and $\|\cdot\|$ is the norm induced by an inner product. According to equation (53) the smaller the value of $J_m(p)$, the better the partitions. Hence the main aim of the criterion is basically to minimize the performance index $J_m(p)$. [15]

Suppose we want to make N clusters $C_1, C_2, ..., C_N$, then the cluster centers $v_1, v_2, ..., v_N$ associated with the partitions would be calculated from

$$v_i = \frac{\sum_{j=1}^{n} [\mu_i(x_j)]^m x_j}{\sum_{j=1}^{n} [\mu_i(x_j)]^m}, \quad i = 1, 2, ..., N \tag{54}$$

where $m > 1$ is a real number that controls the influence of the membership values. [15] Notice here that each calculated vector is indeed a weighted mean for its respective cluster where the weighting is controlled by m. A positive definite matrix $M_{(i)}$ of membership grades formed from objects in each cluster is chosen to initialize the procedure. With an appropriate inner product, a metric and a small number $\epsilon > 0$, cluster centers are calculated using equation (54) and updates made on the matrix at each time. The criterion

is designed to run until the new matrix M_{i+1} is not very different from the previous iteration M_i, thus it terminates when

$$\|M_{(i)} - M_{(i+1)}\| < \epsilon \tag{55}$$

At this point, $M_{(i+1)}$ partitions are suggested, otherwise the process continues. [3]
We have not listed the whole procedure as followed by the criterion for the fuzzy center means algorithm, but equations given in this subsection are essential for understanding other concepts discussed.

3.3.3 Feature selection

The effect of high dimensionality is evidently a problem for one to carry out classification with ease, as discussed earlier in section 3.1, it is always advantageous to know which features are the best for classification. [9] [6] In practice, this is rather difficult because some features may be too complex to analyze and two or more features may have the same characteristics which makes it better to combine them. [29] Feature selection consists of choosing those features which are more effective for preserving data information, some classifiers are sensitive to this stage once it is omitted. [6]

The fact is that some features may be redundant and when removed or combined, the study and computational costs become manageable. [9] There are other benefits of doing the selection like better accuracy since some irrelevant features may lead to loss of accuracy. The major difficulty lay in finding a method of feature selection that will not eventually distort the data or in principle, one would need a method where the features will be invariant to unnecessary transformations which result from the use of some techniques. [29]

3.4 Classification

3.4.1 Fuzzy k-nearest neighbor classifier

The k-nearest neighbor algorithm is used to assign patterns of unknown classification to the class of the majority of its k-nearest neighbors of known classification. [7] A distance measure is key to this algorithm. The classical case assumes that each sample belongs to a specific class but for the fuzzy case there is a possibility of a sample to be in more than one class. [29] The situation could be as in the two cluster data in Figure (7) below where different samples are mixed up. The circled samples are evidently misplaced and for each sample, there is that possibility. For the fuzzy approach, membership values are given to the samples in relation their contribution to the classification process.

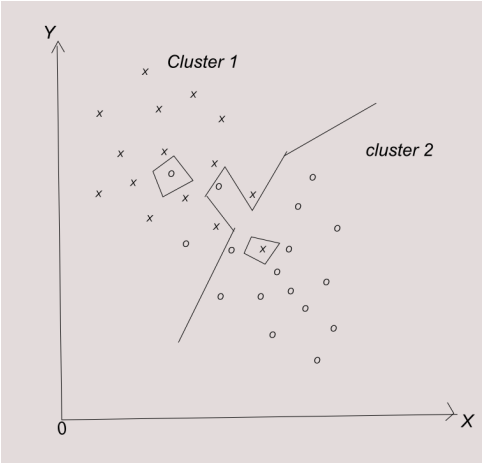


Figure 7: The two cluster data set with some mixed samples.

For the classical case, samples in a particular class are assumed to be equally important and the algorithm may not distinguish between samples as long as they are in the same cluster. [29] However, this is problematic as it is evident in real data that all samples cannot be exactly the same or have the same importance. Hence the fuzzy approach provides a fertile ground for researchers in this field to exploit the realized failures.

The only way to use this classifier in a sensible manner, is to come up with a distance measure that one can use. In this respect, various measures including Euclidean and Mahalanobis distances may be used. Starting from the closest neighbor, the algorithm runs until all the clusters in the neighborhood are covered. Initialization basically determines to which local minima the algorithm will terminate. Thus the algorithm will always search for the labeled patterns in the nearest class. [7], [13]

To compute membership values for different samples, Keller et al.(1985) [13] used the expression below:

$$\mu_i(x) = \frac{\sum_{j=1}^{K} \mu_{i,j}(1/\|x - x_j\|^{2/(n_1-1)})}{\sum_{j=1}^{K}(1/\|x - x_j\|^{2/(n_1-1)})} \tag{56}$$

where $\mu_{i,j}$ is the membership in the i^{th} class of the j^{th} sample of the labeled pattern and n_1 is the scaling factor for fuzzy weights.

Notice that from equation (56), the membership value of a particular sample is largely dependant on the inverse of the distance from its nearest neighbors. Thus much weight is given if the sample is closer and less weight if it is far away from the starting node.

Algorithm for the fuzzy k-nearest neighbor classifier (FKNN)

We present a brief out line of the algorithm, for a details see.[13] Suppose we have a set of m labeled patterns, $W = \{x_1, x_2, ..., x_m\}$, the following procedures are followed:

BEGIN

INPUT x, of unknown classification

Set K, $i \leq K \leq m$

Initialize $i = 0$

DO UNTIL (K nearest neighbors to x are found)

Compute distance from x to x_i

IF $(i \leq K)$ THEN

Include x_i in the set of K-nearest neighbors

END IF

END DO UNTIL

Initialize $i = 0$

DO UNTIL (x assigned membership in all classes)

Compute $\mu_i(x)$ using equation (56)

Increment i

END DO UNTIL

END.

3.4.2 Similarity Classifier

The major issue in classification is to partition the feature space as accurate as possible so that each category goes to where it belongs. [9] However, this is not an easy task to do though the use of distance measures provide a route to follow in classification. [29], [9] Distance measures preferred involves the use of metrics but one can even make use of a non-metric similarity function say $s(x, y)$ to compare two vectors x and y. Similarity values obtained from $s(x, y)$ are large when the two vectors are almost the same and small when vectors are different. In this way, the usual distance measure is understood as a dissimilarity measure. [9]

The task is to find a way to classify a known set of objects X into a required number of different classes say N, we can represent our classes by $C_1, C_2, ..., C_N$. Suppose we know the number of objects say n and the features that we have to measure from these objects say $f_1, f_2, ..., f_n$. We believe that there is some interaction between the objects and it is this interaction that we shall use to fit our classification method.

The appropriate procedure is to make sure that our data is first normalized so that it sits in the interval $[0, 1]$ to facilitate our fuzzy domain. This means that the objects we

shall classify are essentially vectors with values in the unit interval. [19] The next aspect is to compute means for each class across all features, these means will form vectors for each class and we shall call them *ideal vectors*, v_i which we can write for class i as:

$$v_i = (v_i(f_1), v_i(f_2), ..., v_i(f_n)) \tag{57}$$

In situations where the data set is too large with a large number of objects in one class, ideal vectors can be calculated using some samples from each class, but they should be able to represent the class as accurate as possible [9]. If we know some sample X_i that represents class C_i properly, then we could get these ideal vectors as $x = (x(f_1), x(f_2), ..., x(f_n))$. One way to do this would be using the generalized mean, by fixing a parameter say m, vectors are computed from:

$$v_i(d) = (\frac{1}{Y_i} \sum_{x \in X_i} x(f_d)^m)^{\frac{1}{m}}, \quad \forall\ d = 1, 2, ..., n \tag{58}$$

where Y_i is the size of the sample from class i or the number in set X_i.

Ideal vectors play an important role in similarity classification method, because we compare them to the rest of the data in this process. The similarities we talk about are computed between ideal vectors and the data. Generalized Łukasiewicz similarity [21] have been preferred for such comparisons and is basically given by equation (31) but in this case we consider a weight parameter w_d which varies per feature depending on its importance, thus we have:

$$S\langle x, v\rangle = \frac{1}{n} \sum_{i=1}^{n} w_d(1 - |x(f_d)^p - v(f_d)^p|)^{\frac{m}{p}} \tag{59}$$

where $v, x \in [0, 1]$ and p is a parameter used in generalized Łukasiewicz similarity. [21] Notice that if the weights w_d are set to one, then it means that all features are equally important and basically this simplifies the calculation but may reduce accuracy. In experiments with real data, it is important to calibrate the parameters m and p used in the

expression (59) above since they evidently affect the accuracy. Precisely, similarity values increase as the value of p is increased up to when all values get close to 1.
The next example illustrates how the method works for smaller dimensions, for higher dimensions we rely on the computer to facilitate our computational abilities.

Example (3.1)

Consider a data set of eight samples each with five attributes that have been measured and recorded in Table (1). Suppose also that the class to which each sample belongs is known. Let a new sample $x = [42 \;\; 60 \;\; 21 \;\; 10 \;\; 76]$ be introduced, whose class is unknown. Determine in which class this sample belongs using Łukasiewicz similarity.

Table 1: Data obtained from the field for different attributes.

samples	a1	a2	a3	a4	a5	classes
1	59	69	19	4	80	1
2	45	42	20	15	72	1
3	55	66	16	22	56	1
4	30	57	28	5	70	2
5	61	60	33	8	66	2
6	49	54	14	11	64	2
7	44	63	26	13	40	3
8	50	53	17	11	62	3

Solution:

The maximum values for each attribute(down the columns) are $[61, \;\; 69, \;\; 28, \;\; 22, \;\; 80]$ these will be used for scaling our data so that it sits in the unit interval as described in the theory above.

We compute the mean vectors $v1, v2, v3$ for each class from the table as below:

$$v1 = [\tfrac{59+45+55}{3}, \tfrac{69+42+66}{3}, \tfrac{19+20+16}{3}, \tfrac{4+15+22}{3}, \tfrac{80+72+56}{3}] = [53, 59, 18.3, 13.7, 69.3]$$

$v2 = [\frac{30+61+49}{3}, \frac{57+60+54}{3}, \frac{28+33+14}{3}, \frac{5+8+11}{3}, \frac{70+66+64}{3}] = [46.5, 57, 25, 8, 66.7]$

$v3 = [\frac{44+50}{2}, \frac{63+53}{2}, \frac{26+17}{2}, \frac{13+11}{2}, \frac{40+62}{2}] = [47, 58, 21.5, 12, 51]$

Assume that $p = 1$ to simplify the calculation and compute the similarities between the unknown sample and the mean vectors using equation (31):

$S(x, v1) = 1/5(1-|\frac{42}{61}-\frac{53}{61}|+1-|\frac{60}{69}-\frac{59}{69}|+1-|\frac{21}{28}-\frac{18.3}{28}|+1-|\frac{10}{22}-\frac{13.7}{22}|+1-|\frac{76}{80}-\frac{69.3}{80}|) = 0.8914$

$S(x, v2) = 1/5(1-|\frac{42}{61}-\frac{46.7}{61}|+1-|\frac{60}{69}-\frac{57}{69}|+1-|\frac{21}{28}-\frac{25}{28}|+1-|\frac{10}{22}-\frac{8}{22}|+1-|\frac{76}{80}-\frac{66.7}{80}|) = 0.9059$

$S(x, v3) = 1/5(1-|\frac{42}{61}-\frac{47}{61}|+1-|\frac{60}{69}-\frac{58}{69}|+1-|\frac{21}{28}-\frac{21.5}{28}|+1-|\frac{10}{22}-\frac{12}{22}|+1-|\frac{76}{80}-\frac{51}{80}|) = 0.8936$

The new sample has similarity values of 0.8914 with class one, 0.9059 with class two and 0.8936 with class three. The decision to which class the sample belongs is made basing on the largest similarity value. For this example, we conclude that the sample belongs to class two.

3.4.3 Summary of experimental flow

The given data is preprocessed with the use of three algorithms: FRPCA1, FRPCA2 and FRPCA3 each at a time. New data is obtained as linear combinations of the samples from the old data. Matlab codes have been written to execute these algorithms and then the new data proceeds for classification with similarity classifier. Results are obtained for each of the three preprocessing algorithms. A flow chart of the process is presented in Figure (8). We are interested in choosing the method that gives the best classification accuracies when used with the similarity classier.

In the second method, we preprocess the old data as done earlier and then classify the new data using the fuzzy k-nearest neighbor classifier. Results are obtained for all the three preprocessing methods to see which one is the best when used with this classifier. The flow chart of the process is shown in Figure (9).

The final interest is to find which of the two classification methods performs better basing on classification accuracies obtained in both cases.

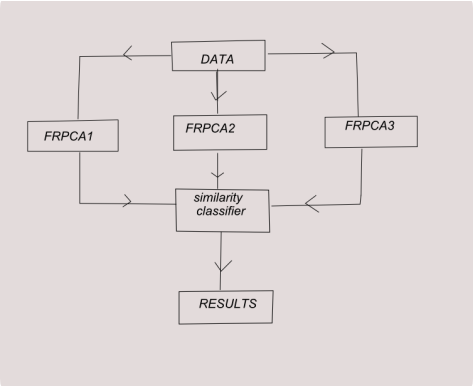


Figure 8: A flow chart for preprocessing and classification with the similarity classifier.

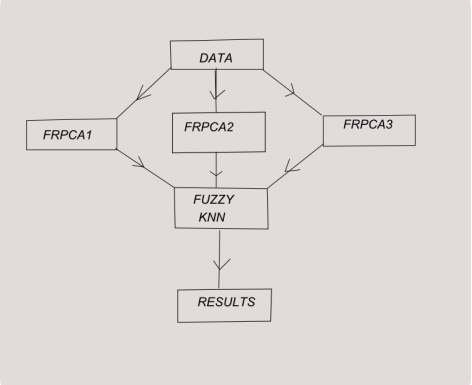


Figure 9: A flow chart for preprocessing and classification with the fuzzy K-nearest neighbor classifier.

4. Classification results

In this chapter, results obtained from experiments are presented. Different sets of results got using different methods are given and described. Emphasis is put on the performance of classifiers as well as preprocessing procedures utilized. Parameter estimation have also been done to obtain acceptable ranges within which parameters in the similarity classifier should lay for this particular data set.

4.1 Results with similarity classifier

In this experiment, as described earlier, ideal vectors are calculated for each class across all features. These ideal vectors are compared with the rest of the samples by calculating the similarities between them. Essentially Łukasiewicz similarity based formula was used for computation of the similarities, [21] the expression used is given as equation (59). Matlab software have been utilized in designing and implementing algorithms from which results are obtained.

4.1.1 Classification accuracy and variance

It is a fact that classification accuracy is affected by data dimension. In this part, we study how many linear combinations that are needed from FRPCA algorithms in order to obtain the best accuracy. Studying the variance gives an insight in concluding how spread the sample accuracies are in relation to the mean accuracy. Different algorithms of FRPCA1, FRPCA2 and FRPCA3 have been used in forming new data with reduced dimension from the old data.

Random division of the data to the training and testing sets is done several times (30 times in this case), mean accuracies and variances are then computed. Classification ac-

curacies relative to data dimension(number of linear combinations used for the new data) have been recorded in Table (2) for a few dimensions and default fuzziness variable (n = 2). More results used in given plots have been put in appendix A for the reader who may need to see numerical performance of other dimensions as obtained in this experiment. Notice that plots have been displayed mostly because it is not easy to analyze results by looking at many values in a table. Plots of such values are helpful in getting a clear distinction between performances of different algorithms. It is observed that classification

Table 2: Classification accuracies obtained using different linear combinations(dimension) from FRPCA1, FRPCA2 and FRPCA3 with the fuzziness variable $n = 2$.

Dimension	FRPCA1	FRPCA2	FRPCA3
9	0.7434	0.7464	0.7506
10	0.7414	0.7329	0.7660
11	0.7509	0.7425	0.8049
12	0.8139	0.7987	0.8089
13	0.8173	0.8184	0.8134
14	0.8127	0.8146	0.8292

accuracies differ for each method used. With only 11 linear combinations, FRPCA3 gives a mean classification accuracy of 80.49%, its performance continues to improve as linear combinations are increased with the best mean accuracy of 82.92% when all features are used. The other two methods also perform well, with FRPCA2 having its best mean classification accuracy of 81.84% and FRPCA1 having 81.73% when 13 linear combinations are used. Over 80% accuracy is still obtained for FRPCA1 and FRPCA3 methods with the first 12 linear combinations.

At this point, it can be seen that FRPCA3 outperforms other methods when the similarity classifier is used for this data set. More experiments will be done on this data to see if these results are conclusive. Notice that results depend on the data set used, thus it is

important to do experiments on each data set of interest. A plot of classification accuracy against dimensions is shown in Figure (10).

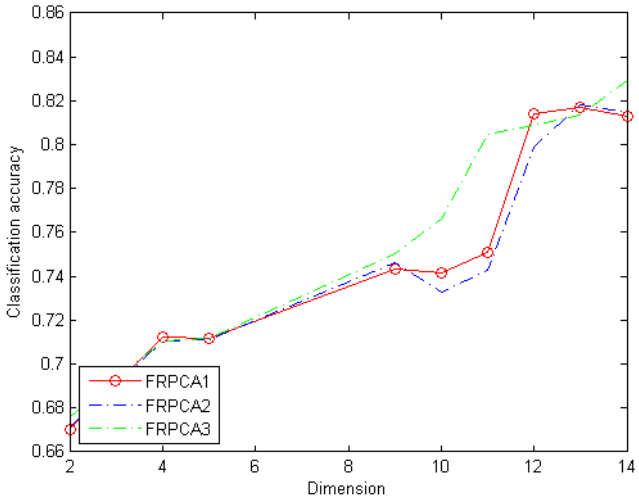


Figure 10: A plot of classification accuracy against dimension for three methods of FRPCA1, FRPCA2 and FRPCA3 with fuzziness variable n = 2.

Different maxima can be seen from Figure (10). Clearly, 13 linear combinations are needed for FRPCA1 and FRPCA2 to perform well but FRPCA3 requires 14 for the best classification accuracy.

Next we look at variances obtained from the experiment for this particular data set. Variance measures the spread of samples in the data in relation to ideal vectors which are the mean reference positions. Notice that obtained variances are small (a few values shown in Table (3)) which suggests that our methods produce acceptable tolerance.

Figure (11) shows the plot of variance against number of linear combinations in the experiment for all the three algorithms. Notice that since variances are small, it indicates that the true classification accuracies are close to the mean values obtained. These variances are helpful in estimating or locating specific accuracy values starting from the mean .

Table 3: Variances obtained for different algorithms and dimension with $n = 2$.

Dimension	FRPCA1	FRPCA2	FRPCA3
10	0.0026	0.0029	0.0027
12	0.0077	0.0077	0.0056
14	0.0067	0.0033	0.0032

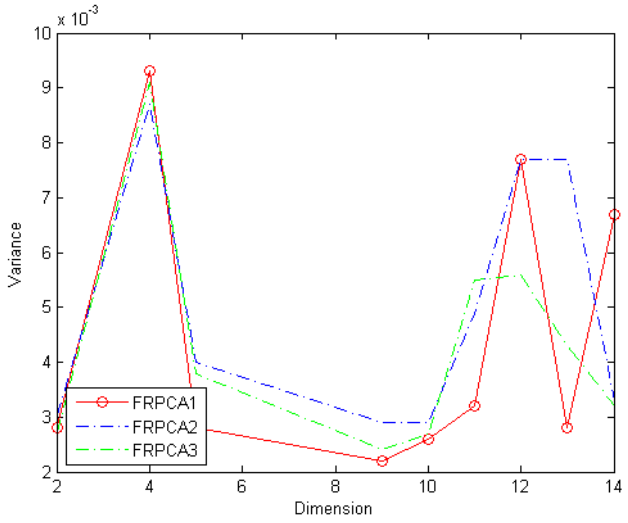


Figure 11: A plot of variance against dimension for three methods of FRPCA1, FRPCA2 and FRPCA3 with n = 2.

This plot indicates that variances obtained using FRPCA3 are more stable.

4.1.2 Changing the fuzziness variable in FRPCA algorithms

The fuzziness variable n was introduced earlier in equation (43) as an important variable that affects classification accuracy. The effect of this variable is different for each data set, that is why we need to study how it affects accuracy for our data set. This has been done by computing classification accuracies for several parameter values. Results of this experiment are shown in Figure (12).

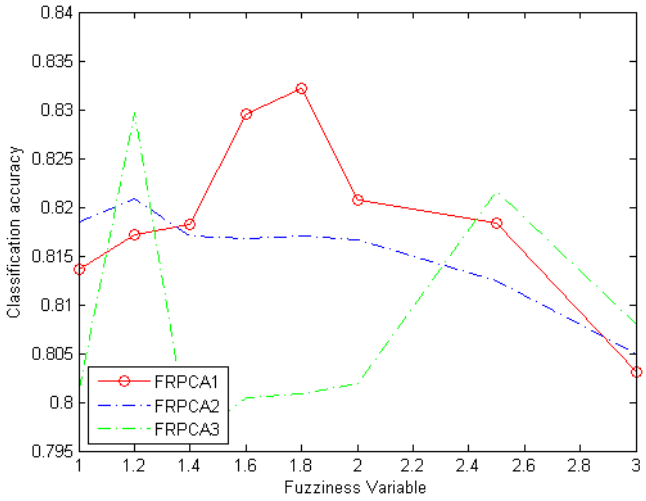


Figure 12: A plot of classification accuracy against fuzziness variable, n with 13 linear combinations from FRPCA1, FRPCA2 and 14 linear combinations from FRPCA3.

From the plot in Figure (12), the best choice of the fuzziness parameter for FRPCA1 is around $n = 1.8$ corresponding to a classification accuracy of 83.22%. With the fuzziness variable around $n = 1.2$, FRPCA2 has a classification accuracy of 82.09% and FRPCA3 has an accuracy of 82.97%. This experiment shows that FRPCA1 algorithm is the best with the similarity classifier when a suitable value of the fuzziness variable is chosen with 13 linear combinations.

4.1.3 Parameters in similarity classifier

Classification accuracies obtained using the similarity classifier relies heavily on correct choice of parameter values in similarity classifier. These parameters m and p were introduced in equation (59) on which these similarities are based. Different data sets require different parameter settings, thus we have to locate suitable values for this particular data

set implemented in this study. One of the easiest ways of locating the best values is to use plots, with this approach we look at contours below the graph of mean classification. Correct choices can be made from the means when plots are clear enough. In this experiment we have considered several cases of linear combinations to produce the best plot from which m and p are estimated. Small linear combinations (8 and less), produced poor results and have not been recorded. Figure (13) below shows results obtained with 10 linear combinations from FRPCA1.

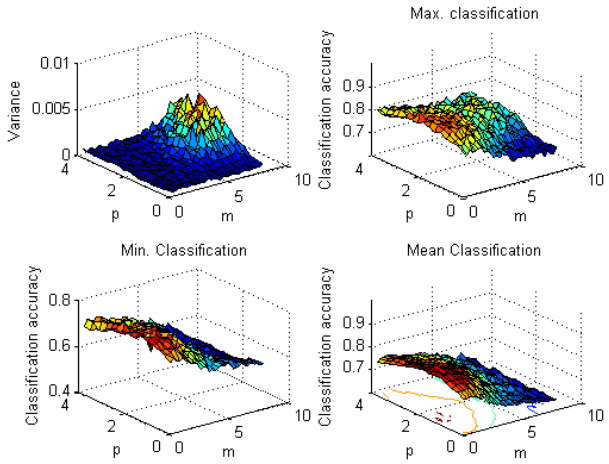
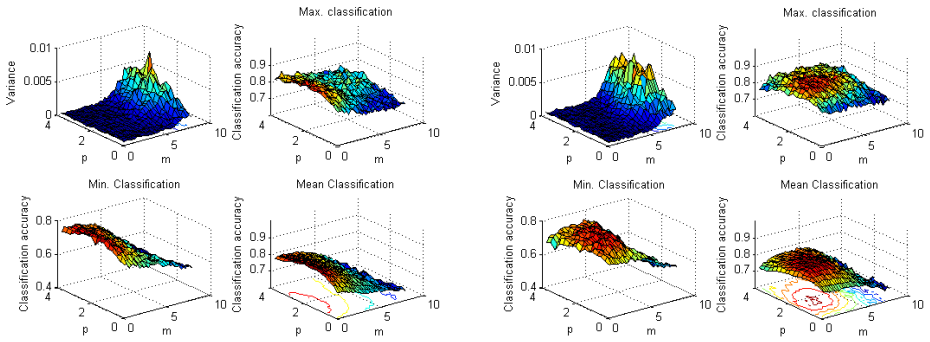


Figure 13: A plot of classification accuracy, p and m for 10 linear combinations from FRPCA1 with the fuzziness variable fixed at $n = 2$.

With 10 linear combinations from FRPCA1 and $n = 1.8$, a classification accuracy of 79.55% is obtained when the values of p and m are around 2.2 and 5.0 respectively. Other FRPCA algorithms give less accuracies with this number of linear combinations. In this experiment, other linear combinations are considered to locate the suitable values of the parameters p and m that produce the best accuracy. When 12 and more linear combinations are used, best results are obtained. Figure (14) shows some of the plots of best results obtained with FRPCA1.

In Figure (14(b)), a mean classification accuracy of 83.22% was obtained using FRPCA1 and 13 linear combinations with $n = 1.8$. Clearly these figures show the intervals where



(a) 12 linear combinations (b) 13 linear combinations

Figure 14: Plots showing the location of best values of the parameters p and m with FRPCA1 when the fuzziness parameter is fixed at $n = 1.8$

parameters p and m should lay. From these plots, it can be seen that the best value of p should be chosen within the interval [0.5, 3], with $p = 2$ seemingly the best choice. The value of m should be taken from the interval [0.3, 6] with $m = 5$ looking best. Thus the best pair for this data set which produced the best accuracy is $(p, m) = (2, 5)$.

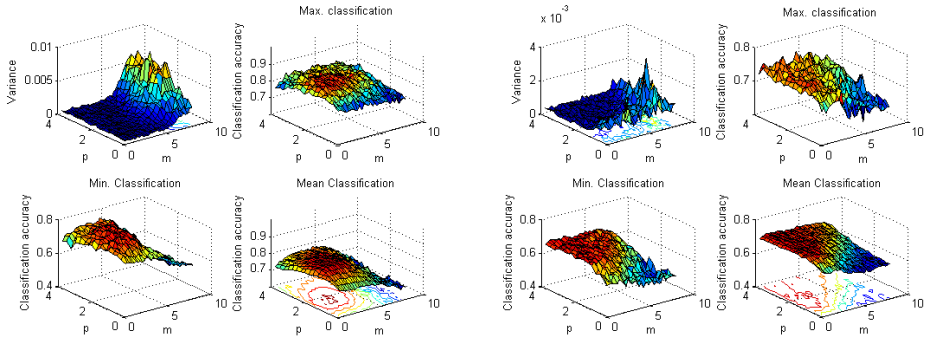
4.1.4 Data allocation to the training set

Classification accuracy is affected by the amount of data assigned for experimentation(training set) depending on the data set used. In this experiment, data is split into the training and testing sets on a 50% basis for each set. However, algorithms have been designed to allocate data to the training and testing sets 30 times on a random basis. We shall study how classification accuracy is affected for this data set and find the best splitting which should be used throughout the whole study. This is done by making other splitting such as 2/3 to the training set which would leave the testing set with 1/3. Results have been found for the training set given by the divisions: $rn = [2/3, 1/2, 1/4, 1/6]$. Using the suitable similarity classifier parameters obtained earlier as $p = 2$ and $m = 5$, a summary of classification accuracies for different FRPCA algorithms are recorded in Table (4).

Table 4: Summary of classification accuracies for different sizes of the training set, FRPCA algorithms, dimension D and fuzziness variable n.

rn	FRPCA1, n= 1.8, D = 13	FRPCA2, n= 1.2, D = 13	FRPCA3, n= 1.2, D = 14
2/3	0.8207	0.8032	0.8071
1/2	0.8322	0.8209	0.8297
1/4	0.7996	0.7819	0.7895
1/6	0.7543	0.7121	0.7335

Accuracy decreases with reduction of data in the training set. From Table (4), the best division is $rn = 1/2$ with the best classification accuracy of 83.22%. In Figure (15), it can be seen that variances are small for large allocations and increases with reduction of the data in the training set. For this data set, $rn = 1/2$ has been used as the allocation for general results recorded in this study.



(a) rn = 1/2 (b) rn = 1/6

Figure 15: Plots showing one of the best data allocation (a) and one of the worst allocation (b) to the training set with 13 linear combinations from FRPCA1 and $n = 1.8$

Looking at Figure (15(b)), the mean classification accuracies obtained with $rn = 1/6$ are generally poor compared to other splitting. The plot in Figure (15(a)) is conclusively good enough for best results but Figure (15(b)) is not good.

4.2 Results with fuzzy knn classifier

In this experiment, two main issues are investigated: linear combinations that should be taken from FRPCA algorithms and the number of nearest neighbors that should be chosen for the FKNN classifier to give the best classification accuracy with this data set. There are two different parameters that are fixed; the fuzziness parameter n in FRPCA algorithms and the scaling factor for fuzzy weights n_1 introduced in equation (56). These parameters affect classification accuracy, thus other experiments are performed to investigate this effect.

4.2.1 Classification accuracy results

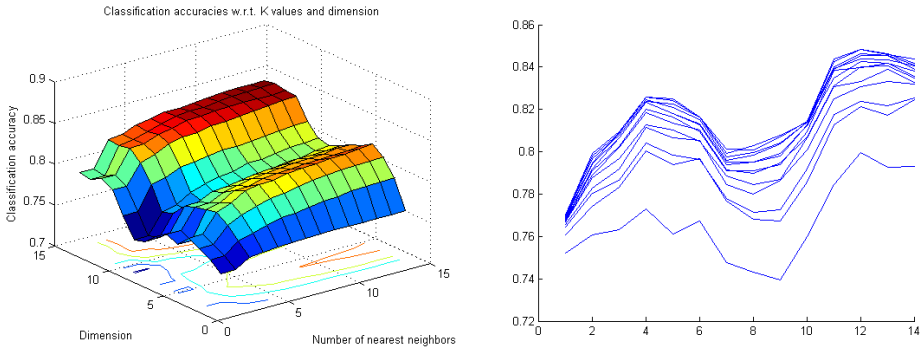
The question of how many linear combinations from FRPCA algorithms required in this case is approached by considering all the three algorithms of FRPCA1, FRPCA2 and FRPCA3. It can be noticed that the FKNN classifier produces considerably more accurate classification accuracies for this data. Table (5) shows some of the best results obtained using FRPCA2 with this classifier for larger linear combinations and relatively many nearest neighbors. More numerical results are put in Appendix A.

Table 5: Classification results using FRPCA2 with fuzzy knn classifier for dimensions D and number of nearest neighbors K when the fuzziness variables $n = n_1 = 2$.

D,K	8	9	10	11	12	13	14
14	0.8366	0.8380	0.8402	0.8414	0.8421	0.8417	0.8427
13	0.8392	0.8390	0.8419	0.8437	0.8437	0.8446	0.8455
12	0.8390	0.8410	0.8413	0.8425	0.8420	0.8418	0.8433
11	0.8344	0.8351	0.8363	0.8381	0.8395	0.8410	0.8422
10	0.8104	0.8101	0.8120	0.8138	0.8153	0.8167	0.8173

Notice that from Table (5) the best classification accuracy is 84.55% obtained with 13 linear combinations and 14 nearest neighbors. These results were obtained with the fuzziness variable $n = 2$ which is the default, better results were obtained after the calibration of n. It can generally be seen that accuracy improved with increase in both dimension and nearest neighbors.

Classification accuracies obtained are well above 81% for 8 and above nearest neighbors when 10 and above linear combinations are used. Figure (16) shows plots for both 3 and 2 dimensional views obtained with FRPCA1. These results depend on which FRPCA algorithm used, thus it will be investigated more to find out which algorithm is the best to use for this classifier and data set.



(a) Classification accuracy, dimensions, number of nearest neighbors

(b) Classification accuracy, dimensions

Figure 16: Plots of classification accuracy for several linear combinations from FRPCA1 with FKNN classifier and the fuzziness variables $n = n_1 = 2$.

From Figure (16(b)), larger linear combinations give better accuracy. At this point, 12 linear combinations gave the largest classification accuracy of 84.62% at $n = 2$ when FRPCA1 was used with the FKNN classifier, but this is not conclusive. Next we study how the fuzziness variable n, in FRPCA algorithms affects the performance of FKNN classifier.

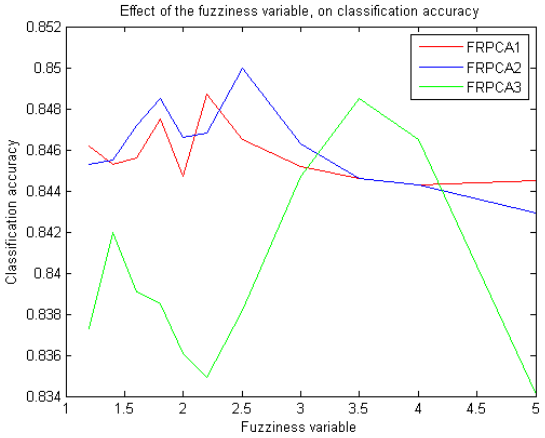
4.2.2 Effect of the fuzziness variable n in FRPCA algorithms

The fuzziness variable n in FRPCA algorithms was studied earlier for the similarity classifier. With the use of the FKNN classifier, it is essential to investigate how this variable affects classification accuracy for this particular data set. This is done by computing classification accuracies for several values of n. Table (6) shows some of the results obtained. Notice that there is another weight parameter n_1 in FKNN classifier introduced in equation (56), which affects classification accuracy. This will be studied later, but for this experiment, a fixed value was used.

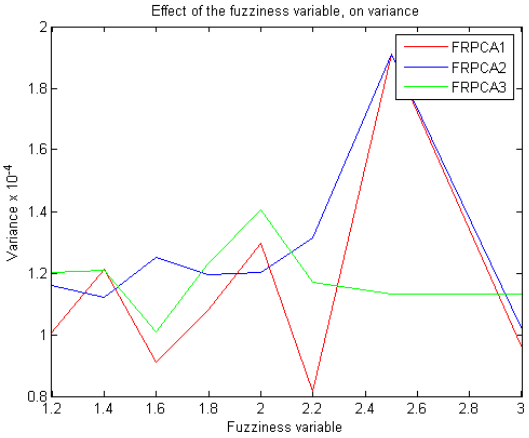
Table 6: Classification accuracies for several values of the fuzziness variable, n in FRPCA algorithms using FKNN classifier with the parameter $n_1 = 2$, 13 linear combinations and 14 nearest neighbors.

n	FRPCA1	FRPCA2	FRPCA3
1.8	0.8475	0.8485	0.8385
2.0	0.8447	0.8466	0.8361
2.2	0.8487	0.8468	0.8349
2.5	0.8465	0.8500	0.8382
3.0	0.8452	0.8463	0.8447
3.5	0.8446	0.8446	0.8482

The best classification accuracy of 85% was obtained with FRPCA2 when $n = 2.5$. With FRPCA1, a classification accuracy of 84.87% was obtained when $n = 2.2$. Lastly FRPCA3, gives its largest classification accuracy of 84.82% when $n = 3.5$. More classification accuracy results together with their variances are shown in Tables (14), (15) and (16) in appendix A. Figure (17) shows the plots of classification accuracy against fuzziness variable (a) and variance against fuzziness variable (b) for all FRPCA algorithms.
Notice that variances obtained in this experiment are small. This means that true clas-



(a) Classification accuracy with fuzziness variable



(b) Variance with fuzziness variable

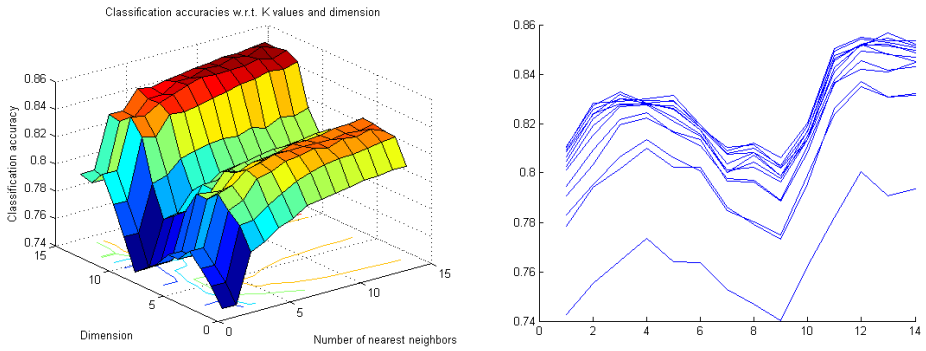
Figure 17: Plots of classification accuracy and variance against fuzziness variable n with the parameter $n_1 = 2$, 13 linear combinations and 14 nearest neighbors.

sification accuracy values are so close to these mean values obtained for this data set. The best choices of the fuzziness variable n in FRPCA algorithms are used for the next experiment where the effect of the parameter n_1 in FKNN classifier is investigated. Hence n is fixed at 2.2, 2.5 and 3.5 for FRPAC1, FRPCA2 and FRPCA3 respectively. Some of the best classifications are shown in Table (7).

Table 7: Classification accuracies for several values of the parameter, n_1 in FKNN classifier with values of n fixed at 2.2, 2.5 and 3.5 for FRPCA1, FRPCA2 and FRPCA3 respectively.

n_1	FRPCA1	FRPCA2	FRPCA3
2.0	0.8481	0.8485	0.8491
2.8	0.8515	0.8552	0.8565
3.0	0.8527	0.8535	0.8566
4.0	0.8516	0.8531	0.8579
4.5	0.8521	0.8593	0.8574
5.0	0.8546	0.8544	0.8562

From Table (7), the best classification accuracy achieved is 85.93% when $n_1 = 4.5$ and $n = 2.5$ from FRPCA2. FRPCA3 produces its largest accuracy of 85.79% when $n_1 = 4.0$ and $n = 3.5$. Finally, FRPCA1 produces its largest accuracy of 85.46 when $n_1 = 5.0$ and $n = 2.2$. Beyond $n_1 = 5.0$ classification accuracies do not improve. For the best parameters in FRPCA2, the next plot shows the general behavior for a number of linear combinations and nearest neighbors.



(a) Classification accuracy, dimensions, number of nearest neighbors

(b) Classification accuracy, dimensions

Figure 18: Plots of classification accuracy for several linear combinations from FRPCA2 with FKNN classifier and the fuzziness variables $n = 2.5,\ n_1 = 4.5$

It is evident from Figure (18) that 13 linear combinations from FRPCA2 and 14 nearest neighbors produce the best classification accuracy of 85.93%. This is obtained when the parameters n and n_1 are fixed at 2.5 and 4.5 respectively.

5. Discussions

In this chapter, research findings obtained in several experiments are summarized. Experiments with similarity classifier showed that FRPCA1 works well with this classifier giving the best mean classification accuracy of 83.22% when $n = 1.8$, and p & m are chosen around 2 and 5 respectively. This accuracy was obtained when 13 linear combinations were used. The FKNN classifier produced best results when paired with FRPCA2, with a classification accuracy of 85.93%. This was obtained when 13 linear combinations, 14 nearest neighbors and fuzziness parameters $n = 2.5,\ n_1 = 4.5$ were used.

It was observed that a classification accuracy above 80% was achieved when FRPCA3 and similarity classifier were used with the first 11 linear combinations. FRPCA1 reached above 80% with 12 linear combinations and FRPCA2 with 13 linear combinations when used with the similarity classifier. For the FKNN classifier, a classification accuracy above 81% was realized for all FRPCA methods when 11 linear combinations with 3 and above nearest neighbors are chosen. This indicates that the FKNN classifier generally performs well compared to the similarity classifier. The best mean classification accuracies got by two different classifiers for all FRPCA methods are summarized in Table (8). More details are provided in Table (9) with all parameters affecting this accuracy.

Table 8: A summary of the best mean classification accuracies obtained with the two classifiers for all FRPCA methods.

Method	similarity classifier	FKNN classifier
FRPCA1	0.8322	0.8546
FRPCA2	0.8209	0.8593
FRPCA3	0.8297	0.8579

The fuzziness variable n affected classification accuracy for the two classifiers used with all FRPCA methods. Results show that each method requires a suitable choice of n depend-

ing on the classifier used. Table (9) shows the best choices for the fuzziness parameter when implementing the data set used in this experiment.

Parameters p and m used in similarity classifier as given by equation(59) were also calibrated. These affect classification accuracy when using the similarity classifier depending on the data set. In this study, an interval was located from which p and m should be picked. Results obtained earlier showed that p should be chosen in the interval [0.5 3.0] with $p \approx 2$ looking best. The value of m should be taken from [0.3 6.0], with $m \approx 5$ preferably the best choice when 13 linear combinations from FRPCA1 are used. All results obtained in the whole study are summarized in Table (9).

Table 9: A summary of the best choices of the fuzziness variables n & n_1, parameters p & m, dimension D and number of nearest neighbors K for two classifiers with all FRPCA methods.

Method	similarity classifier	FKNN classifier
FRPCA1	p $\approx$ 2, m $\approx$ 5	$n_1 = 2.2$ K=14
n	1.8	2.2
D	13	13
Accuracy	0.8322	0.8546
FRPCA2	p $\approx$ 1.9, m $\approx$ 3.5	$n_1 = 4.5$ K=14
n	1.2	2.5
D	13	13
Accuracy	0.8209	0.8593
FRPCA3	p $\approx$ 2.1, m $\approx$ 2.5	$n_1 = 4.0$ K=14
n	1.2	3.5
D	14	14
Accuracy	0.8297	0.8579

6. Conclusions

6.1 Conclusion and recommendation

In this thesis we have found out that it is important to first preprocess the Australian credit card screening data set before classification is done. Preprocessing removes unwanted(outliers) samples and suitable linear combinations are obtained to be used as the new data. For the similarity classifier, this should be done using the FRPCA1 algorithm with 13 linear combinations and the fuzziness parameter set to $n = 1.8$. Parameters in the similarity classifier should be taken around $p \approx 2$ and $m \approx 5$. The best procedure for classification based on this data set is to use 13 linear combinations from FRPCA2 with FKNN classifier and 14 nearest neighbors. The fuzziness parameters n and n_1 should be 2.5 and 4.5 respectively. This produced the best classification accuracy of 85.93% in the whole study. The choice of linear combinations and the number of nearest neighbors for the FKNN approach are essential for this data set. The most exciting part of this approach is the way the algorithm is able to test several k-values simultaneously with ease rather than a single scaler in its neighborhood.

6.2 Future Work

The next task of this research could be optimization using fuzzy techniques. In the new challenge, a global optimizer such as a differential evolution algorithm is chosen and implemented. The method aims at automatically optimize parameter values in the used classifier, fuzziness parameter in FRPCA algorithms and the choice of FRPCA method. Another already developed area uses non-linear FRPCA algorithms. This can also be tested and developed further. Hybrid robust PCA can be developed where probabilistic approach is combined with fuzzy PCA in applications.

Bibliography

[1] Atiya, A. F. (2001). Bankruptcy prediction for credit risk using Neural Networks: A new survey and new results, IEEE transactions on neural networks (Vol. 12(4)), pp 929-935. Accessed online as pdf at: http://www.ru.lv/p̃eter/ida/bankruptcy/bankr.pdf

[2] Bandemer, H., Nather, W. (1992), Fuzzy Data Analysis, Kluwer Academic Publishers Group, 322,3300 AH Dordrecht, The Netherlands, Pages 61-176.

[3] Bezdek, J. C. (1981). Pattern Recogniton with Fuzzy Objective Function Algorithm, Plenum, New York.

[4] Charmi, P. (2011). Solving Leontief Input-Output Model with Trapezoidal Fuzzy Numbers and Gauss-Seidel Algorithm, Master's thesis, Lappeenranta University of Technology.

[5] Chennakesava, A. R. (2006). Fuzzy logic and Neural Networks: Basic concepts and applications, New age International (P) limited, New Delhi-India, Pages 6-18.

[6] Chih, F. T. (2009). Feature selection in Bankruptcy prediction, An imprint of Elsevier Science, Knowledge-Based Systems Vol. 22(2009), Pages 120-127.

[7] Chung, F., Rhee, H., Hwang, C. (2003). An Interval Type-2 Fuzzy K-Nearest Neighbor, Hanyang University, Ansan-Korea, IEEE 0-7803-7810, Pages 802-807.

[8] De Luca, A., Termini, S. (1972). A definition of a non probabilistic entropy in the setting of fuzzy set theory, Information control 20, pp 301-312.

[9] Duda, R. O., Hart, P. E., Stork, D. G. (2001). Pattern Classification, Second edition, A Wiley-Interscience publication, ISBN 0-471-05669-3 USA, Pages 168-199.

[10] Frank, A., Asuncion, A. (2010). UCI Machine Learning Repository, Irvine, CA: University of California, School of Information and Computer Science. Accessed online at: http://archive.ics.uci.edu/ml

[11] Jolliffe, I. T. (1986). Principal Component Analysis, Springer - Verlag, Inc 175 Fifth Avenue,New York, Ny10010, USA.

[12] Keinosuke, F. (1990). Introduction to Staistical Pattern Recognition, Second edition, A Harcourt Science and Technology Company, San Diego USA.

[13] Keller, J., Gray, M., Givens, J. R. (1985). A fuzzy K-Nearest Neighbor algorithm, IEEE Transactions system, (Vol. 15(4)), pp 258-263. Accessed online as a pdf at: http://people.cs.missouri.edu/škubicm/8820/FuzzyKNN.pdf

[14] Kagan, T., Ghosh, J. (1999). Linear and Order Statistics Combiners for Pattern Classification, Pages 1-25.

[15] Klir, G. J., Yuan, B. (1995). Fuzzy sets and Fuzzy logic: Theory and Applications, Prentice Hall Inc., USA, Pages 97-175.

[16] Kumar, P. R., Ravi, V. (2007). Bankruptcy prediction in banks and firms via statistical and intelligent techniques-a review, European Journal of operational research, (Vol. 180(1)), Pages 1-28.

[17] Luukka, P. (2009). PCA for fuzzy data and similarity classifier in building recognition system for post-operative patient data, Expert systems with applications, 36(2P1), Pages 1222-1228.

[18] Łukasiewicz, J. (1970). Selected works, Cambriadge University press.

[19] Luukka, P. (2009). Classification based on fuzzy robust PCA algorithms and similarity classier, Expert systems with applications, 36(2009), Pages 7463-7468.

[20] Luukka, P. (2005). Similarity measure based classification, Doctorial Thesis, Lappeenranta University of Technology.

[21] Luukka, P.,Saastaminen, K., Knnen, V. (2001). A classifier based on the maximal fuzzy similarity in the generalized Łukasiewicz structure, Fuzz-IEE, Melbourne, Australia.

[22] Min, J. H., Lee, Y. C. (2005). Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters, Expert Systems with Applications, Vol.28(2005), pp 603-614. Accessed online as pdf at: http://hompi.sogang.ac.kr/bsscom/min05_1.pdf

[23] Negnevitsky, M. (2002). Artificial intelligence: A guide to intelligent systems, Biddles Ltd, Guildford and king's Lynn, Great Britain, Pages 87-128.

[24] Nguyen, T. H., Elbert, A. W. (2000). A first course in Fuzzy logic, Second edition, Chapman and Hall/CRC United States of America, Pages 61-78.

[25] Nock, R., Elomaa, T., Kaariainen, M. (2003). Reduced Error Pruning of branching programs cannot be approximated to within a logarithmic factor, Information Process, Lett. vol 87, Page 73. Accessed online as pdf at: http://www1.univ-ag.fr/řnock/Articles/Drafts/ipl03-nek.pdf

[26] Rountree, N. (2006). Initialising Neural Networks with Prior Knowledge, Ph.D. Thesis, Doctor of Philosophy, University of Otago, New Zealand. Accessed as pdf at: http://www.cs.otago.ac.nz/research/publications/OUCS-thesis2-2007.pdf

[27] Shin, K. S., Lee, Y. J. (2002). A genetic algorithm application in bankruptcy prediction modeling, Expert Systems with Applications, Pages 321-328.

[28] Sugeno, M. (1974). Theory of fuzzy integrals and its applications, Ph.D. Thesis, Tokyo Institute of Technology, Tokyo.

[29] Theodoridis, S., Koutroumbas, K. (1999). Pettern Recognition, An imprint of Elsevier Science, 525 B Street, Suite 1900, San Diego, California 92101-4495, USA, Pages 139-155, 565-584.

[30] Xu, L., Yuille, A. L. (1995). Robust principal component analysis by self organizing rules based on statistical physics approach, IEEE Transactions on Neural Networks, Vol.6(1), pp 131-143. Accessed online as pdf at: http://www.cs.cuhk.hk/l̃xu/papers/journal/IEEENN95.PDF

[31] Yager, R. R. (1979). On the measure of fuzzines and negation, part 1: Membership in the unit interval, General Systems,Vol. 5, Pages 221-229.

[32] Yang, T. N., Wang, S. D. (1999). Robust algorithms for principal component analysis: Pattern recognition letters, Vol. 20, pp 927-933. Accessed online as pdf at: http://ntur.lib.ntu.edu.tw/bitstream/246246/142280/1/17.pdf

[33] Yi-Cheng, H. (2003). Learning control knowledge for planning, Ph. D. Thesis, Doctor of Philosophy, Faculty of the Graduate School of Cornell University.

[34] Zadeh, L. (1965). Fuzzy sets, Information and control,8, Pages 338-353.

[35] Zhang, G., Hu, M. Y., Patuwo, B. E., Indro, D. C. (1999). Artificial neural networks in bankruptcy prediction: general framework and cross-varidation analysis, European Journal of operational research, Vol. 116(1999), pp 16-32. Accessed online as pdf at: http://www.agsm.edu.au/bobm/teaching/SimSS/zhang.pdf

Appendix A

Some experimental results

Table 10: Classification accuracies obtained using the similarity classifier for several values of the fuzziness variable, n and all FRPCA methods.

n	FRPCA1	FRPCA2	FRPCA3
1.1	0.8136	0.8184	0.8011
1.2	0.8172	0.8209	0.8297
1.4	0.8182	0.8171	0.7959
1.6	0.8296	0.8168	0.8004
1.8	0.8322	0.8171	0.8009
2.0	0.8207	0.8166	0.8019
2.5	0.8184	0.8125	0.8216
3.0	0.8031	0.8049	0.8080

Table 11: Mean, maximum(Max.sf), minimum(Min.sf) classification accuracies and variances obtained from 30 classification runs with different linear combinations for all FRPCA methods using the similarity classifier.

Method	Dimension	mean	Max.sf	Min.sf	Variance
FRPCA 1	14	0.8127	0.8642	0.7896	0.0067
FRPCA 2	14	0.8146	0.8647	0.7890	0.0033
FRPCA 3	14	0.8292	0.8757	0.8121	0.0032
FRPCA 1	12	0.8173	0.8699	0.7803	0.0028
FRPCA 2	12	0.8184	0.8699	0.7803	0.0077
FRPCA 3	12	0.8134	0.8526	0.7717	0.0043
FRPCA 1	5	0.7119	0.7630	0.6850	0.0028
FRPCA 2	5	0.7109	0.7659	0.6821	0.0040
FRPCA 3	5	0.7117	0.7630	0.6821	0.0038
FRPCA 1	4	0.7121	0.7630	0.6821	0.0093
FRPCA 2	4	0.7109	0.7717	0.6792	0.0087
FRPCA 3	4	0.7119	0.7746	0.6850	0.0091

Table 12: Classification results obtained by FRPCA2 with FKNN classifier for different linear combinations(D) and number of nearest neighbors(K). The fuzziness variables were set to $n = n_1 = 2$ as the default.

D,K	1	2	3	4	5	6	7
14	0.7910	0.7910	0.8222	0.8241	0.8310	0.8329	0.8373
13	0.7886	0.7886	0.8203	0.8236	0.8283	0.8328	0.8376
12	0.7958	0.7958	0.8233	0.8242	0.8314	0.8336	0.8389
11	0.7882	0.7882	0.8120	0.8177	0.8253	0.8296	0.8302
10	0.7633	0.7633	0.7913	0.7955	0.8033	0.8052	0.8080
9	0.7395	0.7395	0.7633	0.7682	0.7788	0.7842	0.7885
8	0.7434	0.7434	0.7683	0.7676	0.7744	0.7781	0.7830
7	0.7558	0.7558	0.7831	0.7839	0.7917	0.7933	0.7969
6	0.7648	0.7648	0.7883	0.7888	0.7971	0.8004	0.8051
5	0.7580	0.7580	0.7909	0.7950	0.8048	0.8060	0.8102
4	0.7724	0.7724	0.7948	0.8017	0.8117	0.8139	0.8152
3	0.7658	0.7658	0.7843	0.7931	0.7982	0.8032	0.8050
2	0.7581	0.7581	0.7714	0.7766	0.7812	0.7857	0.7872

Table 13: Classification results obtained by FRPCA2 with FKNN classifier for different linear combinations(D) and number of nearest neighbors(K). The fuzziness variables were set to $n = n_1 = 2$ as the default. Continuation from Table(11) with higher values of K.

D,K	8	9	10	11	12	13	14
14	0.8366	0.8380	0.8402	0.8414	0.8421	0.8417	0.8427
13	0.8392	0.8390	0.8419	0.8437	0.8437	0.8446	0.8455
12	0.8390	0.8410	0.8413	0.8425	0.8420	0.8418	0.8433
11	0.8344	0.8351	0.8363	0.8381	0.8395	0.8410	0.8422
10	0.8104	0.8101	0.8120	0.8138	0.8153	0.8167	0.8173
9	0.7906	0.7959	0.7960	0.7983	0.8007	0.8016	0.8033
8	0.7858	0.7893	0.7910	0.7935	0.7939	0.7980	0.8000
7	0.7988	0.8007	0.8014	0.8038	0.8039	0.8041	0.8060
6	0.8065	0.8075	0.8092	0.8103	0.8107	0.8141	0.8126
5	0.8125	0.8133	0.8143	0.8155	0.8157	0.8164	0.8185
4	0.8184	0.8202	0.8210	0.8222	0.8230	0.8245	0.8266
3	0.8086	0.8112	0.8129	0.8148	0.8158	0.8160	0.8162
2	0.7895	0.7917	0.7933	0.7954	0.7961	0.7964	0.7974

Table 14: Mean classification accuracy and variance for several values of the fuzziness variable(n) with $n_1 = 2$ when FRPCA1 is used with the FKNN classifier.

n	Mean accuracy	Variance x10^{-4}
1.2	0.8462	1.0043
1.4	0.8453	1.2125
1.6	0.8456	0.9122
1.8	0.8475	1.0777
2.0	0.8447	1.2970
2.2	0.8487	0.8165
2.5	0.8465	1.1909
3.0	0.8452	0.9591
3.5	0.8446	1.1033

Table 15: Mean classification accuracy and variance for several values of the fuzziness variable(n) with $n_1 = 2$ when FRPCA2 is used with the FKNN classifier.

n	Mean accuracy	Variance x10^{-4}
1.2	0.8453	1.1603
1.4	0.8455	1.1195
1.6	0.8472	1.2520
1.8	0.8485	1.1944
2.0	0.8466	1.2017
2.2	0.8468	1.3135
2.5	0.8500	1.1907
3.0	0.8463	1.0174
3.5	0.8446	1.0155

Table 16: Mean classification accuracy and variance for several values of the fuzziness variable(n) with $n_1 = 2$ when FRPCA3 is used with the FKNN classifier.

n	Mean accuracy	Variance x10^{-4}
1.2	0.8373	1.2022
1.4	0.8420	1.2102
1.6	0.8391	1.0101
1.8	0.8385	1.2310
2.0	0.8361	1.4041
2.2	0.8349	1.1714
2.5	0.8382	1.1333
3.0	0.8447	1.1330
3.5	0.8482	1.1645

Table 17: Mean classification accuracies for several values of the parameter n_1 in FKNN classifier for all FRPCA methods with 13 linear combinations and 14 nearest neighbors.

n_1	FRPCA1, n = 2.2	FRPCA2, n = 2.5	FRPCA3, n = 3.5
1.2	0.8205	0.8098	0.8142
1.6	0.8377	0.8426	0.8388
2.0	0.8481	0.8485	0.8491
2.4	0.8495	0.8467	0.8552
2.8	0.8515	0.8552	0.8565
3.0	0.8527	0.8535	0.8566
3.5	0.8540	0.8557	0.8570
4.0	0.8516	0.8531	0.8579
4.5	0.8521	0.8593	0.8574
5.0	0.8546	0.8544	0.8562

Table 18: Variances for several values of the parameter n_1 in FKNN classifier for all FRPCA methods with 13 linear combinations and 14 nearest neighbors.

n_1	$\times 10^{-4}$ FRPCA1, n = 2.2	$\times 10^{-4}$ FRPCA2, n = 2.5	$\times 10^{-4}$ FRPCA3, n = 3.5
1.2	1.1432	1.2298	1.2428
1.6	1.0700	1.2291	1.3408
2.0	1.0777	1.2347	0.8677
2.4	1.1761	1.1607	0.9640
2.8	0.7650	1.1242	1.0273
3.0	0.8740	1.1737	0.9352
3.5	0.8909	0.9847	1.0047
4.0	0.9777	1.1326	0.9659
4.5	1.2010	1.0135	1.0462
5.0	1.1618	1.1622	1.1391